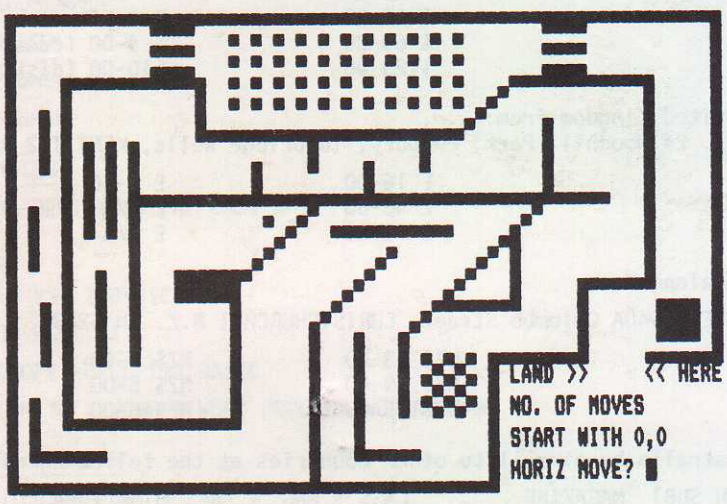
• TRS-80 • SYSTEM 80 • VIDEO GENIE
• PMC-80 • HITACHI PEACH
• TRS-80 COLOUR COMPUTER

Vol. 3, Issue 11, October 1982



MOVE BY 1'S

Also in this issue:

PROGRAMMING:
Making Decisions the Easy Way

REVIEWS:
The LNW 80 Mark II
ACCEL3 vs. ZBASIC

HARDWARE:
Computer Anatomy—Part I

SOFTWARE:
- BIORHYTHMS—Colour
- CHECKSUM—Level II
- CODEBREAKER—Level II
- FLEXITIME—Level II
- INPUT DEMONSTRATION—Level II
- LOAN CALCULATION—Colour
- MATRIX MANIPULATION—Level I

MICRO-80

P.O. BOX 213, GOODWOOD, S.A. 5034. AUSTRALIA. TELEPHONE (08) 211 7244. PRICE: AUS. $2.50, N.Z. $4.00, U.K. £1.50
MICRO-80 is registered by Australia Post — Publication SQB 2207 Category B

***** ABOUT MICRO-80 *****

EDITOR:	RYSZARD WIWATOWSKI
ASSOCIATE EDITORS:	
SOFTWARE :	CHARLIE BARTLETT
HARDWARE :	EDWIN PAAY

MICRO-80 is an international magazine devoted to the Tandy TRS-80 Model I, Model III and Colour microcomputers, the Dick Smith System 80/Video Genie and the Hitachi Peach. It is available at the following prices:

	12 MONTH SUB.	SINGLE COPY
MAGAZINE ONLY	$ 26-00	$ 2-50
CASSETTE PLUS MAGAZINE	$ 65-00	$ 4-00 (cass. only)
DISK PLUS MAGAZINE	$125-00	$10-00 (disk only)

MICRO-80 is available in the United Kingdom from:
U.K. SUBSCRIPTION DEPT. 24 Woodhill Park, Pembury, Tunbridge Wells, KENT TN2 4NW

MAGAZINE ONLY	£ 16-00	£ 1-50
CASSETTE PLUS MAGAZINE	£ 43-60	£ N/A
DISK PLUS MAGAZINE	£ 75-00	£ N/A

MICRO-80 is available in New Zealand from:

MICRO PROCESSOR SERVICES, 940A Columbo Street, CHRISTCHURCH 1 N.Z. Ph 62894

MAGAZINE ONLY	NZ$ 43-00	NZ$ 4-00
CASSETTE PLUS MAGAZINE	NZ$ 89-00	NZ$ 5-00
DISK PLUS MAGAZINE	NZ$175-00	NZ$15-00

MICRO-80 is despatched from Australia by airmail to other countries at the following rates:

(12 MONTH SUB)	MAGAZINE	CASS + MAG	DISK + MAG
PAPUA NEW GUINEA	Aus$40-00	Aus$ 83-00	Aus$ 143-00
HONG KONG/SINGAPORE	Aus$44-00	Aus$ 88-00	Aus$ 148-00
INDIA/JAPAN	Aus$49-00	Aus$ 95-00	Aus$ 155-00
USA/MIDDLE EAST/CANADA	Aus$55-00	Aus$102-00	Aus$ 162-00

Special bulk purchase rates are also available to computer shops etc. Please use the form in this issue to order your copy or subscription.

The purpose of MICRO-80 is to publish software and other information to help you get the most from your TRS-80, System 80/Video Genie or Peach and its peripherals. MICRO-80 is in no way connected with any of the Tandy, Dick Smith or Hitachi organisations.

** WE WILL PAY YOU TO PUBLISH YOUR PROGRAMS **
Most of the information we publish is provided by our readers, to whom we pay royalties. An application form containing full details of how you can use your microcomputer to earn some extra income is included in every issue.

** CONTENT **
Each month we publish at least one applications program in BASIC for each of the microcomputers we support. We also publish Utility programs in BASIC and Machine Language. We publish articles on hardware modifications, constructional articles for useful peripherals, articles on programming techniques both in Assembly Language and BASIC, new product reviews for both hardware and software and we print letters to the Editor.

** LIABILITY **
The programs and other articles in MICRO-80 are published in good faith and we do our utmost to ensure that they function as described. However, no liability can be accepted for the failure of any program or other article to function satisfactorily or for any consequential damages arising from their use for any purpose whatsoever.

***** CONTENTS *****

	PAGE
REGULARS	
EDITORIAL	2
PEEKING (UK) - From our U.K. Correspondent	3
INPUT/OUPUT - Letters to the Editor	5
MICROBUGS	19
SUBMITTING PROGRAMS FOR PUBLICATION	7
DEPARTMENTS	
KALEIDOSCOPE	3
PEACH BOWL	3
GROUP ONE	4
FORM THREE	4
PROGRAMMING	
MAKING DECISIONS THE EASY WAY	9
REVIEWS	
ACCEL3 VS. Z.BASIC	13
LNW 80 MARK II	15
MICROSOFT LEVEL III BASIC	17
DICK SMITH'S WORP-1 WORD PROCESSING PROGRAM	18
HARDWARE SECTION	
COMPUTER ANATOMY - PART 1	11
SOFTWARE SECTION	
LOAN CALCULATIONS.....CC	21 & 26
LOAN CALCULATIONS.....PEACH	21 & 27
BIORHYTHMS.....CC	21 & 26
BIORHYTHMS.....PEACH	21 & 28
MOVE BY 1'S L2/16K	22 & 29
CHECKSUM.....L2/4K-48K m.l.	22 & 31
MATRIX MANIPULATION.....L1/4K	23 & 32
INPUT DEMONSTRATION.....L2/4K	24 & 33
FLEXITIME.....L2/4K	25 & 33
CODEBREAKER.....L2/4K	25 & 34
MICRO-80 PRODUCTS CATALOGUE	CENTRE
NEXT MONTH'S ISSUE	35
CASSETTE/DISK EDITION INDEX	36
ORDER FORM	36

MICRO-80 is registered by Australia Post - Publication SQB 2207 Category B

AUSTRALIAN OFFICE AND EDITOR: MICRO-80 P.O. Box 213, GOODWOOD SOUTH AUSTRALIA, 5034. TEL. (08) 211 7244

U.K. SUBSCRIPTION DEPT: 24 WOODHILL PARK, PEMBURY TUNBRIDGE WELLS, KENT TN2 4NW

Printed by: Shovel & Bull Printers, 379 South Road, MILE END 5031

Published in Australia by: MICRO-80 433 Morphett Street, Adelaide.

MICRO-80 has developed a Library of Software consisting of 7 programs and a comprehenisve user manual. The Software Library, on cassette, will be sent FREE to every new subscriber and to every subscriber who renews his subscription for another 12 issues. Disk subscribers will receive their Software Library on a diskette. The Software Library contains Level I and Level II BASIC and machine language programs.

***** EDITORIAL ******

NEW LOOK MICRO-80

Starting with this issue, MICRO-80 has a new look. We have introduced a number of specialised departments to cater for the diverging interests of our readers and to allow us to publish much more detailed information about the four major computer systems which we support:

- Model I (including TRS-80, System-80, Video Genie and PMC-80)
- Model III
- TRS-80 Colour Computer
- Hitachi Peach

These four departments will be featured in each issue. In addition there will be a number of other specialised departments which will appear regularly covering:

Printers
Disk Drive Systems
Programming
Software and Hardware Reviews
Hardware Projects

Of course we will continue to publish lots of new software each month just as we have in the past and will answer readers' letters and problems.

We hope you like our new look and more importantly, the extra information you will find each month for your particular system. Naturally, we are seeking articles to fit in with our new format for which we will, of course, pay publication fees. So, why not get straight to work preparing articles, hints, tips, etc. about your favourite computer, printer and disk drive system and send them in to us.

HUMBLE PIE DEPARTMENT

Whilst sweeping away all the old cobwebs we have decided to make a further change. All of our readers are well aware that the date shown on the cover is about six months behind the actual date of issue. This is a legacy of our earlier days of more limited resources and, try as we will, we have not been able to catch up without sacrificing quality. What we have been able to do is to hold our own and produce a magazine each month for the past three issues. Well, you had better dig out those calendars that came with the July issue because we have decided to put aside our pride and to change the cover date so that the magazine you receive will carry the actual date of issue. We must allow time for two printing processes, since we print the magazine in the United Kingdom after we print it in Australia. Therefore, the very next issue of MICRO-80 magazine will carry a cover date of JULY 1983. It will still be volume 3, issue 12 however and all our subscribers will receive the number of issues to which they are entitled. You can rely upon us to inform you when your subscription needs renewing as we send out a reminder notice with the second last issue of your subscription.

THE KING IS DEAD! LONG LIVE THE KING!

Until now, the undisputed king of the dot matrix printers has been the MX-80. Epson has dropped the MX-80 from its product line and has introduced two new models - the FX-80 and the RX-80. The FX-80 is at the top of the range featuring high speed printing (160 c.p.s.), a host of new capabilities and a classic styling that borrows much from its predecessor. The RX-80 looking even more like the MX-80 also provides improved performance. The price of the RX-80 is about the same as the MX-80 ($995 including tax) while the FX-80 is rather more expensive ($1399 including tax).

Before you begin to mourn the passing of the MX-80 too deeply, there are rumours and strong evidence that indicate Epson is now licensing other manufacturers to produce MX-80 look-alikes. We have already sighted two such copies from different manufacturing and suspect that soon the market will be saturated with them. These printers feature modest improvements in capability over the MX-80 and are priced at around $700 including tax.

SPECIAL DELIVERY

I am happy to report that the response to our expanded Input/Output column has been tremendous. So many letters are coming in that it would be impossible to publish them all in the magazine. We are doing our best to answer them all, some in the magazine, others by individual reply. I would particularly like to thank the many readers who have written offering help with other readers' difficulties and I look forward to seeing much more of this in the future.

- 0000000000 -

***** PEEKing (UK) - by Tony Edwards *****

The sale of personal computers is still booming in the U.K. Sales have tripled in the last nine months to a total of 111,000 in February 1983. February was also the end of the first 12 month period when more than 1 million micro-computers were sold in this country. The figure of installed micro-computers in the U.K. is now given as 1.1 million, which does not say much for the staying power of computers if 1 million are to be sold each year.

Of the new computers sold, 68% were built in the U.K. and only 8% in the Far East, although these figures conceal the fact that a lot of the pre-assembly work for 'UK' machines takes place in the Far East. In terms of new home computers, the first three positions are unchanged, with Sinclair as market leader, followed by Commodore and Acorn. Tandy is now down to 9th position with 7,300 units delivered in the three months up to February '83.

Of installed units, as would be expected, Sinclair leads the field with 560,000 installed units with Tandy at 7th with 46,000 and Lowe (who market the Genie) being 9th with 35,000 units.

The continued increase in sales appears to have taken a number of manufacturers by surprise as the demand for a number of machines still far exceeds the supply, and many buy the machine which is available at the time of purchase, rather than wait for delivery which may take months.

- 0000000000 -

***** KALEIDOSCOPE *****

This month our programming article (Making Decisions the Easy Way) should be of special interest to our Colour Computer readers and I do hope to see some programs using the techniques described appearing in the near future. As well, we have another two programs that have been converted by editorial staff to run on the Colour Computer - Loan Calculation and Biorhythms.

We have had some inquiries from our readers regarding the possibility of offering a cassette edition for the Colour Computer. Unfortunately, at this stage we can be no more definite than to say we are looking in to it. At the moment, the main concern is to keep the cost as low as possible and although, the recording cost would not change, the tape testing cost, which is labour intensive, would be appreciably higher.

We came across a program in 80-Micro (that's the American publication with which we sometimes get confused) that would allow the use of four colours in high resolution mode, as opposed to the specified two. However, it was disappointing to find, after entering the program, we still had only two. This does not necessarily mean there's a bug in the program. In fact, it should have been expected as programs of this type usually depend upon the television system used which is NTSC in the U.S., while here in Australia we use the PAL system.

Here's a tip for disk users. There is a simple way of producing printed copy of a directory listing. If you enter the following:

POKE 111,254:DIR (Enter)

then the directory will be sent to the printer.

- 0000000000 -

***** PEACH BOWL *****

Last month, we published the first program for the Peach actually written by a reader, and I'm happy to say, we have a few more on the way. So you can look forward to these in future issues. This month we have two more converted programs we hope you will find interesting - Loan Calculation and Biorhythm.

Despite the fact that the Peach is a Hitachi product, the owners have found that support, particularly from the manufacturer, has been less than adequate. However that situation has been remedied as NISSEI SANGYO ADVANCED PRODUCTS (a subsidiary of Hitachi) has created a division to control distribution and support of Hitachi small computer products in Australia. As well as announcing three new Peach models, new peripherals like the Pocket Floppy 3" disk are now available. The Level 3 price has dropped to $1,195, while the new Level 4, featuring 64K RAM, CP/M and PEACHSOFT will cost you $1,495. A step further is the Level 5, featuring MS/DOS and 128K RAM, for $1,795 but the price of the Level 6 with 64K RAM, FLEX and OS-9, is yet to be announced. Together with a small selection of software, including the three most popular games, full professional documentation is a standard feature. (Prices do not include tax).

Have you tried to modify a BASIC program only to find that it has been 'Protected' and you cannot LIST it? Well, there is a simple way to overcome this feature. After loading the program do the following:

(1) MON (Return) - to enter the monitor.
(2) MB4 (Return) - to display the protect-byte; it should be $FF (hexadecimal)
(3) Ø (Return) - change it to zero
(4) Control D (Return) to exit to BASIC

You should now be able to LIST the program. If you get nothing at all (as if there were no program present), then the program has probably been UNLISTed as well as protected. To fix that, you must go back into the monitor and set the bytes at $117 and $118 to $FF (all in hexadecimal), returning to BASIC via 'control D'.

- 0000000000 -

***** GROUP ONE *****

This month we present no less than five programs for Level II users and one Level I program that has been adapted by the original author. As well, the hardware and software reviews are pertinent and, I hope, will satisfy those readers who have recently asked about some of these products. Such information derived from actual users is the best kind when you are thinking about buying.

It is good to see a healthy response to some reader's pleas for help although for practical reasons, only two have been published. There have been other situations where users have managed to solve their problems or overcome their difficulties, because the magazine has access to a large resource - all of our subscribers.

Don't overlook the programming article simply because examples for the Colour Computer are referred to - the techniques and ideas presented are equally relevant to Level II users, and with a little modification can be readily applied.

There is some more news on LDOS. I can remember stating that there was no mention of the floppy disk versions for the Models I and III, but, thanks to one of our readers, I must make amends. The floppy disk versions are, in fact, quite clearly included in the Tandy catalogue (although not where I would have expected) and are now available for $179.95.

- 0000000000 -

***** FORM THREE *****

Naturally, much of what applies to the Model I is also applicable to the Model III. Rather than repeat what is said above, I instead refer you to it and point out the differences here.

Many Level II BASIC programs will work on the Model III without any modification whatsoever and I forsee no difficulty with the four published in this issue. Unfortunately, the Checksum machine language program will not work on the Model III.

When Tandy upgraded from Model I to Model III, changes were made to the ROM software - particularly in the low address range below 760H. Extra ROM was added to provide for enhancements in both software and hardware, but for the most part, the actual BASIC interpreter is almost the same as the Model I version. Also, to accommodate these extra features, more scratchpad RAM has been allocated to the tune of 256 bytes. This is the main source of incompatibility between the Model III and Model I.

For some hints on how to overcome some of these problems, I refer you to the Micro-bug relating to the MOVIE utility. Perhaps, that section should have been included, more appropriately, here as there is nothing wrong with the utility at all - it just doesn't work on the Model III. The Checksum utility could be adapted to work on the Model III if it were relocated and changed in a similar way to the MOVIE utility. It must be relocated because the BASIC line input buffer is located in a different place on the Model III - at 42E8H instead of 41E8H as on the Model I.

- 0000000000 -

***** INPUT/OUTPUT *****

From: Mr. G. Farmer - South Tamworth, N.S.W.

I am writing to you to request some information. Firstly I will explain my position. I have just purchased a System 80 computer so that I can gain some programming experience to help me with my Electrical Engineering Degree. This is great for BASIC, but I would also like to be able to program in PASCAL. I've tried the 'Tiny Pascal' as supplied by Tandy and Dick Smith but this program seems to be very unstable as well as not covering the full Pascal language that I need for my course. I have not been able to track down any disk versions of the language in the same line as the Cobol and Fortran languages available for the TRS-80 or System 80. Do you know of any such packages available in Australia (or overseas), that can be obtained? If so, how suitable would they be to an electrical engineering application (i.e. do they cover the language sufficiently?). Your help on this matter would be most appreciated.

(The only version of which I am aware is Pascal-80 by New Classics Software but there may be others. Pascal-80 is compatible with most of the DOSes available for the Model I, and is a reasonably comprehensive implementation, although not complete. For full details, I refer you to an advertisement in 80 Micro, February 1983 (p.138). Unfortunately, I know of no similar product available in Australia. Perhaps some other reader knows of other implementations? - Ed.)

From: Mr. C. De Silva - Greystanes, N.S.W.

I have a TRS 80 III 65K in which when I try to hold memory from 15360 to 17686 I get an "OM ERROR". I can hold up to 65529 (though 65K is = 66560) missing 1041 bytes . Do I have a bad Ram (2326 bytes) and if so can you tell which one? When asked to print MEM the display is 48082, though I have a 65K. What's wrong?

(What seem to be memory problems in your Model III are, in fact, normal responses and there is no cause to suspect faulty memory. The initializing routine in the ROM (Read Only Memory) on power up does not allow you to set the top of BASIC's memory below 17686 or above 65529. However, to clarify the situation,the maximum available user RAM (Random Access Memory) memory in the Model III is 48K. I refer you to your manual which contains information explaining how the address space is assigned in the Model III. - Ed.)

From: Mr. J.A.Wardley - Auckland, N.Z.

Three letters in one week to the same address, in my world, is in itself a record I can assure you. However, I have just realised that you may well be able to help me with a problem that I have with my Hitachi MB-6890 and the connection to the Epson MX-80F/T III printer.

My Epson has the RS-232C/Current Loop Interface Type 2. The problem occurs when I try to print anything that is greater than the 2K buffer in the printer, everything after this is lost to the printer.

At present the machines are set as follows: Baud rate is set at 4800, CS3 in the MB 6890 is set to 3. CS4 is set to 7, 6, 4, 2.

Any help you may be able to offer would be very much appreciated.

(Unfortunately, this problem cannot easily be cured because the problem lies in the software ROM routines that service the RS-232-C interface. These do not respect the handshaking lines - in particular, the Clear To Send line. The only ways to avoid character loss are to either slow down the transmission rate (not guaranteed to work) or to write your own replacement routines. Alternatively, you may opt for using the parallel interface which does observe the Busy signal. - Ed.)

From: Mr. M. Kelloway - Barrack Point, N.S.W.

I'm having a problem with a program I got out of my first issue of Micro-80. The program I'm referring to is in Volume 3 No. 8 (July 1982) called "Dr. Who Adventure". I keep getting an FC error in line 74Ø and being a 'beginner', I don't know how to overcome this problem.

I have a Dick Smith System 80 (16K) computer. I hope you can help me to get this program to run.

(Line 74Ø as listed in your letter (omitted above) is correct and should not cause any error to occur. The Function Call error results, in this particular case, from passing a null character (or no character) to the ASC function. This can happen if any of the strings associated with the variables SO, SP or SD are too short, as may happen if these strings are not read correctly from cassette tape in line 73Ø.

What I suggest you try is to make another copy of the data tape by running the Initialiser again and see if that cures the problem. Apart from this, I'm at a loss to find another cause. Maybe other readers can offer some other suggestions? - Ed.)

From: Mr. D. Mollica - Doncaster East, Vic.

Whilst trying to load "Dr. Who Adventure" into our 16K TRS-80 I kept getting an "OM ERROR" every time the tape counter showed 229 (I was loading the second copy), whereas the program actually stopped at 237. However, a "CLEARØ" executed before loading provided that extra little bit (50 bytes) of memory to enable "Dr. Who" to fit into 16K.

(A number of readers have called or written with a similar problem. This problem only manifests itself when you attempt to load the main program after having run the Initialiser. The reason for it seems obvious after a little experimentation.

The Initialiser CLEARs 5000 bytes of string space for its own use. However, although CLOAD does a NEW to remove the existing program, it does not reset the string space allocated back to 50 bytes - the default value on power up. So the result is that only some 10,622 bytes of memory are free to CLOAD the "Dr. Who Adventure", which is too little and results in the Out of Memory error.

You can check the amount of string space allocated at any time by typing PRINT FRE(X$), and the amount of free memory by typing PRINT FRE(X) or PRINT MEM. Since the "Dr. Who Adventure" requires nearly all of the 16K of memory it is necessary to type CLEAR 5Ø after the Initialiser has been run (or CLEAR Ø as suggested by Mr. Mollica) to reclaim the memory allocated to strings.- Ed.)

From: Mr. K. Dare - Hillman, W.A.

First let me say that your magazine is, by far, one of the best magazines for '80 users I have seen. However I have noticed a disturbing trend in some of your more recent issues finally coming to a head with your Morse Practice program. Recently, it seems, you have taken to the practice of giving us the hex dump only and omitting the source code listing, meaning that those of us who like to see programs for new techniques in programming, or like to modify or relocate the code for any reason (e.g. adding the use of a joystick) find this extremely difficult.

I realize that space is at a premium within your magazine, but for the reasons above, and also the possibility of a typeset error which can be detected and repaired far more easily with a source code listing, I think that readers will appreciate a full listing to get full value from a program.

With regards to the Morse Practice program the hex listing supplied is in the same memory area as TBUG and ZBUG, the two monitors I have at my disposal. This was a program I am very interested in, but unable to use, so if possible could we see the full source code listing for this and future programs.

(Although I agree in principle, the practice of omitting source code listings for extremely long programs shall continue, simply for the reason that they do consume too much space. It seems to me to be more unreasonable to devote over 20% of the space in the magazine to one program. However, this does not mean to imply that we will deny access to the source code of long machine language programs to those readers who wish to study them.

In fact, a source code file is provided on both the cassette and disk editions of the magazine and these can be purchased at the single unit prices indicated on the inside front cover. For those who do not have access to a printer, copies of the listing can be obtained for the nominal charge of $5.00 to cover copying costs and postage (however, this depends on the size of the listing and the current rates).

The problem of memory contention between your monitors and the Morse Practice program, although apparently insurmountable at first, can be overcome with the Z80's powerful block move instructions. As a suggestion, you could load the program into the memory from 7394H to 77A1H and append the following block move code:

77A2: 21 94 73 11 94 43 Ø1 ØE Ø4 ED BØ C3 94 43

Start: 7394H End: 77AFH Entry: 77A2H

This will move the block of memory down to the correct location and commence executing it. This technique will also suit the method of using a BASIC program to load machine language programs into low memory normally used by the program itself. - Ed.)

From: Mr. Ron Hack - Reynella, S.A.

Following the three requests for help with the Asylum Adventure (MICRO-80 Sept. '82) I have listed some hints avoiding the actual answers as this could spoil the game for others.

SECOND GUARD

Remember what you were told when caught by the first guard. Also, it's too late if you have already seen the second guard.

FIRST MAZE

If you find yourself unable to escape from this maze then you have not fully explored it. As the instruction pamphlet suggests, you would do well to draw the maze as you go, remembering at certain parts you will be teleported to other parts.

INMATE WITH COATHANGER

We cigarette smokers have no trouble here. Don't forget the instruction pamphlet's suggestion to examine things.

OTHER HINTS

Walk around (and draw) the administration block until you fully understand it.

Murderers can read too.

You will notice no doors are locked in the twenty door corridor.

Some objects are used more than once.

Finally, I found it helpful to make up a separate list of all the objects from the vocabulary list and cross them off as I found them, thereby knowing what I still had to find.

Good luck and happy adventuring.

From: Tony Mason - Grange, S.A.

Your September '82 issue had pleas from three desperate Asylum inmates asking for a few clues. I'm no expert on Asylum, but I can tell you it is a difficult adventure..... I have at least three witnesses to attest to it!

I know you're all dying to know your solutions, so here they are:

To Robbie Thomas: You are by no means alone. This is the hardest problem you will face in your quest for freedom.

The Clue: You must trick the guard. This involves WALKING (as no inmate would be so audacious as to WALK the corridors) around corners until you meet the guard, and then HITting him.

To J. Terhoeve: Getting out of the maze requires a key. This key is guarded by a crazed axe murderer (the computer gives you clues from there). If you haven't yet found the part of the maze where he appears, I recommend you make a map - it's a big maze.

To B. Ronning: I also got stuck here, having tried 'everything'. The only thing, I'll bet, you haven't tried yet, is TRADE. A few hints here..... that inmate has a taste for Rothmans and a sense of humour.

And before you all go rushing off to try them..... the pick is hidden on the maze on Peladon.

As I have nothing to do all day but solve adventures and help others do the same, I am also available for Zork I, The Institute, Labyrinth, and Mission Impossible. Remember, I was stuck in them once, too.

- 0000000000 -

***** SUBMITTING PROGRAMS FOR PUBLICATION by Charlie Bartlett *****

As the software editor for MICRO-80, you don't normally hear much from me unless you actually submit a program, so I thought it was time that you did. If you have written a program and would like to share it with the other readers of the magazine, then the place to send it is:

MICRO-80 SOFTWARE DEPARTMENT
P.O. BOX 145,
MORPHETT VALE, S.A. 5162.

Some of you will notice that this address is different from the business and postal addresses of MICRO-80 as indicated on page one. Why am I explaining this? Well, you see the software department is physically quite a long way away from the Morphett Street office in the city. So if you are sending in a program for publication, you should send it here because if you don't, then you will increase the length of time before receiving a reply, as MICRO-80 will have to send it here before anything happens. Similarly, if you send Mail Orders here, I will have to send them there before they get processed.

In view of the fact that a lot of new subscribers have joined us (and for the benefit of some of the old ones who may have forgotten), I would like to take this opportunity to mention a few of the DO's and DON'Ts of submitting software that will help both you and me when it comes to processing your programs.

1. Your submission should include a copy of the program on machine-readable media in standard format, a description of how to use the program and some explanation of how it works. All

should be clearly marked with your name and address (it is also a good idea to include your name and address in REM statements at the beginning of the program). The term 'standard format' is defined as follows:

(a) TRS-80 Model I/III and System 80:

Cassette - 500 baud
Disk - 35 or 40 track, single-sided single density

(b) TRS-80 Colour Computer:

Cassette - Standard
Disk - 35 Track, single-sided, double-density

(c) Hitachi Peach:

Cassette - Standard
Disk - 40 track, single-sided, single-density

2. Please use good quality media (tape and disk) since, although every effort is made to load a program, after about a dozen tries we give up and move on. Your original media will be returned to you upon request, provided you have included return postage.

3. Save yourself a stamp, don't write in and say, "I have this program that does so and so. Would you be interested?" Until I see it, I cannot tell - be brave and send in the program in the first place. We have never shot anyone for sending in a program that we didn't like.

4. Please don't send in just a listing of the program as time just does not permit me to sit there and type it in, so it will only be returned with a request that it be supplied on tape or disk.

5. To make your program useful to as many people as possible, it should be compatible with a minimum system type as follows:

   (a) 16K, cassette based TRS-80 Model I/III and System 80.

   (b) 16K, cassette based Extended Colour BASIC TRS-80 Colour.

   (c) 32K, cassette based Hitachi Peach.

   However, programs that have specific requirements should explain these clearly in the accompanying description.

WHAT TO EXPECT WHEN YOU SUBMIT A PROGRAM

First of all, please don't expect an immediate reply by return mail. We have several hundred subscribers who regularly submit programs and it takes a while to go through them. The length of time that it takes me to reply varies, depending on the quantity of incoming mail, but I usually try to return a reply in under a month.

When your program comes to the top of the pile the very first thing that happens is that it is copied onto a file disk and your paperwork is filed. The program is then severely tested with a deliberate mental approach designed to try to make it fail. No-one likes to type in a long program from the magazine only to find it doesn't work properly. Any "minor" flaws found at this time are usually fixed right then and there. The program is then evaluated on several points:

(1) Does the program do what it is supposed to do?

(2) How original is the program?

(3) Will it appeal to enough of our subscribers?

Depending on the type of program, other factors can come into the final decision. Any help you can provide in this testing phase will save time and speed up the process. For example, all machine language programs should be accompanied by the source code, preferably on tape or disk. Adventure type programs should be accompanied by a solution to help validate their correctness. If you are unlucky, you will receive a form which has been designed as a compromise between not having the time for individual replies and fulfilling the need to assist you, the author. We try to avoid decreasing your confidence, but rather strive to encourage you to try again. After all, a rejection is something from which you can learn.

Now, happy days, this is what happens if your program is accepted. You will receive a letter along with a form headed, Temporary Assignment of Copyright. The assignment form gives us permission to publish your program, for which we pay a publication fee. After publication, the program is once again your property.

Now, although we have accepted your program for publication, it does not mean that your program will appear in the very next issue. I have several months of Level II programs on hand and it could be some time before it appears in the magazine. We try to balance the contents of the magazine by including games, utilities and general interest programs. However, as we receive

more games than anything else, it stands to reason that if you send in a game, you will have to wait a lot longer to see it published than if you send in a utility. Also, we have programs from different authors on exactly the same subject. Where both programs have something of merit in them, we will accept both of them for publication. Now, of course, when we publish the first one, the second author thinks that he has been forgotten. This is not the case. We don't want to publish two programs of the same type too close together for obvious reasons. So, the second program is held for a while and published at a later date. If you have submitted a program and have since seen a program like yours published you may think we have forgotten you. Relax, we have you on file and it will get published.

However, authors of Level I programs and colour computer programs for the Tandy and the Peach are more fortunate at the moment, due to the lack of programs.

By the way, please don't rush out and start spending the money that we offered. Remember, we don't pay for your program until after we publish it - after all, that's what we are paying for, the right to publish.

(It's funny, I think I can hear those keyboards rattling already!)

Well, there it is. I hope you are lucky enough to receive an acceptance. If not, well, just keep on trying and if you have never submitted a program, do it now! You have NOTHING to lose but a stamp and the possibility of a lot of personal satisfaction in making your computer earn you some money (no fortune, mind you, but then it's a crazy thought anyway, actually getting paid for something you love doing anyway).

- 0000000000 -

***** MAKING DECISIONS THE EASY WAY!!?? by Richard Newcombe *****

This item was really provoked by the obvious need in computer games for large amounts of logic to make the multitude of decisions necessary as compared to pure number crunching one may find in other programs. Although some of the examples given here apply to some features of the colour computer, the general use of IF...THEN...ELSE statements is common to many computers and languages other than BASIC. The inherent logic in this particular BASIC statement is fairly self evident but its method of actual function lends itself to far more versatile use when properly understood. In the following discussion the basic syntax of the command will be discussed in the form as follows.

IF statement THEN action1 ELSE action2.

Implicitly if the 'statement' is TRUE then 'action1' is computed otherwise if the statement is FALSE then 'action2' is acted upon. Big deal you say!! The next part may seem hard and more complicated than it really is, but it is important not to read more into it than actually exists. Essentially, if a statement is calculated as being FALSE the computer actually registers the value zero for the statement, if the statement is TRUE the value of the statement is designated minus one (forget about the minus sign for the moment, as it is more of a nuisance value than anything else). As an example, examine the following program lines.

```
10  X=3:Y=5
20  Z=(X>Y)
```

As X is, in fact, less than Y, line 20 is evaluated as FALSE and Z is given the value of 0.

The main intent of the following manipulations is really to produce rather more concise programming and also varying degrees of increased program execution speed, which is particularly important during action orientated games. I must mention at this point that this form of logic can actually be slower for one particular reason - in an equation (of operators), every part of the statement has to be evaluated. On the other hand, an IF-THEN-ELSE statement can skip the ELSE if the initial statement is TRUE (this is probably more significant speed-wise if the conditional statements are nested). One major advantage of this method is the ability to use conditional operations in DEF FN functions. Conciseness is an advantage but the speed issue is one with which you will have to experiment.

In the next example a conditional IF statement is shown in line 10 with the following line 20 producing exactly the same results.

```
10  IF X>Y THEN Z=Y ELSE Z=0
20  Z= -(X>Y) * Y
```

Line 10 needs little comment so let's focus on line 20. Here we have an expression in which (X>Y) is tested and if TRUE is given the value minus one (-1). The change of sign is compensated

for by the preceding minus which inverts the value and the resultant value Z is computed from Y multiplied by 1.

If X was less than Y then the section in brackets would have the value of zero and as a result (0)*Y would have the value zero (and hence also Z).

Now that we are armed with the knowledge that FALSE is equal to zero and TRUE is equal to minus one we can almost forget about the IF statement.

Now it may be more useful to look at a practical example which was really intended for use with the colour computer. However, understanding that JOYSTK is a command that reads the joystick inputs, the body of the following lines should not be too hard to understand (!!??). Here the intent was to ignore the normal values of the joysticks which can range from zero to 63 (here we were only interested in left, centre and right positions).

Such that the joystick should not be too sensitive to centre position, left was defined as a value less than 20 and right a value greater than 45. After these values were determined they had then to be converted into an increment or decrement to shift an object about the screen. The next step was one of determining whether or not an attempt was being made to move the object beyond the bounds of the screen, and if such was the case, then no action should be taken. After the listing I will describe each line by number.

Note that INC (i.e. increment) =-8 as the sign change is made to allow for TRUE=-1 (same applies to other values).

```
10   INC=-8: DEC= 8: LM=7: RM=229
800  J1= JOYSTK(0)                                  'get joystick value
820  XJ= (J1>45)*INC + (J1<20)*DEC                  'check joystick value
880  XZ= (XJ=8)*(XX<RM)*XJ + (XJ=-8)*(XX>LM)*XJ     '? at L or R margin
940  XX=XX+XZ: YY=YY+YZ                             'increment counter
960  IF XZ OR YZ THEN GOSUB 1020 ELSE 820           '? branch to graphics routine
```

By contrast line 820 could be written as...

```
820  IF J1>45 THEN XJ=INC ELSE IF J1<20 THEN XJ=DEC ELSE XJ=0
```

In line 820 XJ would ultimately be +8, 0 or -8 as follows (i.e. the decrement or increment). If J1 is greater than 45 this component would be TRUE and INC (-8) would be multiplied by minus one. At the same time, and under these circumstances, J1 would not be less than 20 and this component of the formula would be zero. The inverse is also TRUE but if neither the tests of J1 are TRUE (being greater than 45 or less than 20) then in this instance XJ would be equal to zero.

Line 880. The next line was to test whether or not any object was on the boundary of the screen and whether any further motion in that particular direction should be calculated. In this line, if XJ is positive (XJ=8) then motion right would be determined and negative, then motion left would be determined (XJ= -8). If XJ was zero then the whole statement would be calculated as zero. Also by including a test such as (XJ= 8) we can determine under which circumstances certain calculations should follow.

The next test (XX<LM etc.) is of course whether or not we are at the left margin or right margin. Here the point on the screen, as determined by XX, is compared with RM and LM, i.e. the appropriate values on either boundaries of the screen. Note that in this example, as two tests have to be TRUE for XZ to be equal to XJ, two tests result in TRUE * TRUE or -1*-1 which gives us a resultant +1.

Obviously keeping track of this side of the result through this manipulation of -1 can be a little tedious and prone to producing bugs in the program.

Line 940. Here XZ has a value of -8, 0 or +8 and XX is now recalculated and for demonstration I have included YY as this was also used to determine the vertical position (the logic for this was not shown for the sake of simplicity).

Line 960. This next part we see that if a value of zero is given for both of XX and YY then the plotting function in the line 1020 (not shown) is not (unnecessarily) undertaken and thus time wasted. Equally so if we were not interested in YZ then this could be as easily stated as

```
IF XZ THEN GOSUB 1020 ELSE 820        or        IF XZ<>0 THEN GOSUB 1020 ELSE 820
```

Notice that if XZ appears alone then any value other than zero for XZ is calculated as TRUE for the statement and the subsequent action taken. It is unnecessary under these circumstances to include additional operators such as not-equal-to-zero (i.e. <> 0) as this wastes the BASIC interpreter's time.

Hopefully, it is fairly apparent as to how much code can be condensed by using these TRUE or FALSE concepts in producing functions rather than conditional statements (with IF etc.).

This may at first seem a little foreign and for this reason also a little hard, but with a little thought, quite considerable savings can be made on decision-making and subroutine execution. Some game listings I have variously read in such publications as Rainbow, although otherwise fairly efficient, are extremely difficult to read and followed by the multitude of IF THEN ELSE statements and some bugs I am sure can arise from multiple levels of nesting that are so easy to lose track of.

- 0000000000 -

***** HARDWARE SECTION *****

The material that usually appears in this section is either a complete constructional project or some form of modification to computer related hardware. This month we begin a series of articles aimed at trying to increase the hardware understanding of newcomers and those who feel the subject belongs in the 'too-hard' basket. The first part reviews the basic concepts as an introduction.

***** COMPUTER ANATOMY - PART 1 *****

Becoming a more able BASIC programmer and learning the rudiments of assembly or machine language programming requires an understanding of how your computer works (at the hardware level) and of the fundamental hardware in the machine. Many are deterred from this task by the very complexity of the subject and the tendency to use unfamiliar terms in explanations. However, reflect upon your first experiences with your computer when you were probably apprehensive about connecting the various pieces together, frustrated by failure to load the demonstration tape, etc. Most problems can be overcome in time with some application and perseverance.

Perhaps the best way to begin looking at something as complex as a computer is to sub-divide the hardware into smaller functional parts, as shown in Figure 1, and to examine each of these separately.

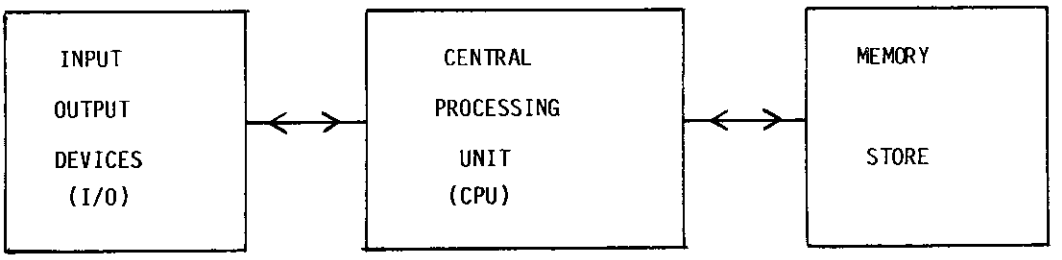


Fig. 1. The Functional Blocks of a Computer

A computer is a machine that is designed to accept input information, process the given information according to some specific instructions and then present the resultant output to the user. Consider the simple hand-held calculator that provides a set of keys for entering numbers, other keys for selecting the required operations to be performed and a numeric display device such as a liquid crystal display (LCD) for presenting the result of computations. It is, in effect, very similar in operation to a computer and can serve to illustrate the functioning of computers in general.

INPUT/OUTPUT

A computer must be able to communicate with its human user and devices are provided to enable this human-machine interface. Perhaps the most obvious input device is the keyboard which is similar to a typewriter keyboard and used to enter information in a similar way, while the usual output device is a T.V. set or a video monitor upon which text consisting of alphabetic, numeric and other characters can be displayed.

However, there are many other devices that can be used to input information to (or receive output from) a computer and these are used in the interests of greater efficiency. For example, a printer can be used to produce a permanent record (hardcopy) of the output. Cassette recorders and disk drives can be used to store (or retrieve) information to (or from) magnetic media such as tape and floppy disk respectively.

Note that these last two mentioned can function as either input or output device whereas the preceding devices can function in only one way: the keyboard as an input, the video and printer as output devices. So we can refine the schematic computer to show more detail as in Figure 2.

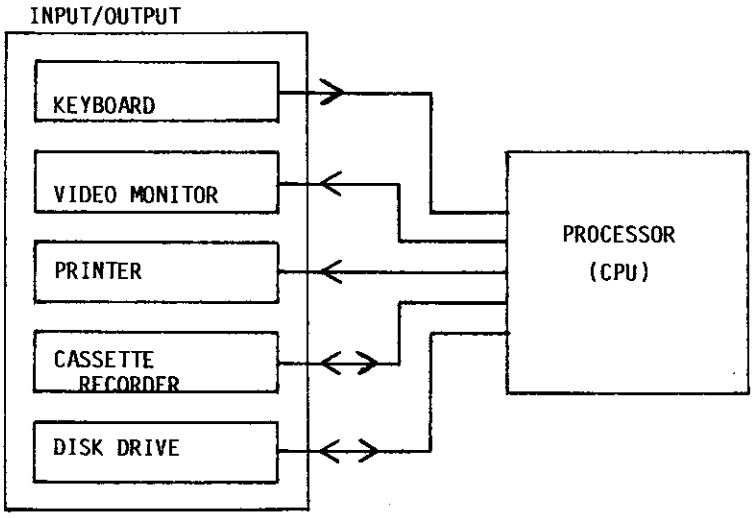


Fig. 2. Details of I/O Devices

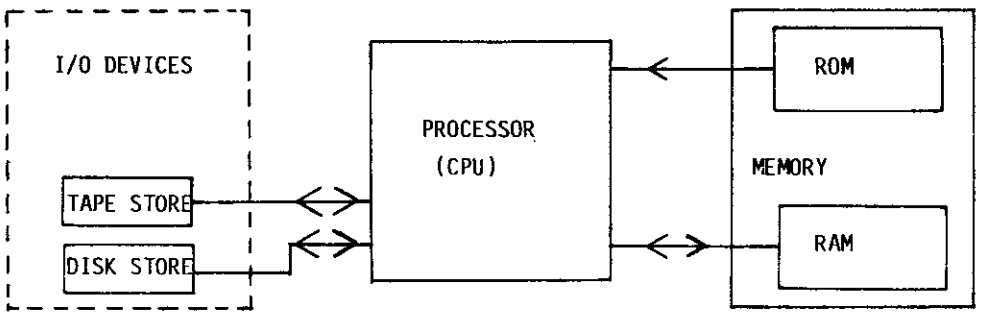
MEMORY

Because computers can process large amounts of information and do so very quickly, it is necessary to provide storage for this information and the processing instructions - this store is called memory. The pocket calculator typically provides a small amount of memory but the computer generally has much more. There are two basic types of memory found in most microcomputers - Read Only Memory (ROM) and Random Access Memory (RAM). The contents of ROM memory is fixed at its time of manufacture and can only be read by the computer as its name suggests. This type of memory is usually used to store tables of data or processing instructions that do not change. A more important feature of ROM is that its contents are permanent and are not lost when power is removed from the computer.

On the other hand, the computer can both read the contents of RAM memory and write different information there if so instructed. However, this read/write memory is volatile in nature and all of its contents are lost when the power is removed. The term 'Random Access' is perhaps inappropriate as ROM memory can also be randomly accessed. The fundamental difference between the two is that the processor can modify the contents of RAM, but not of ROM.

These two types of memory, usually called primary memory, are semi-conductor memories and can operate at reasonably high speed. The two I/O devices mentioned above, the cassette recorder and disk drive, also provide information storage on the media they use and this can be considered as secondary storage (see Figure 3). However, as these devices are external to the computer and mechanical in nature, the time taken to access the information stored is quite long in comparison to the primary memory.

Fig. 3. The Storage Available in the Computer



PROCESSOR

The basic function of the processor is to follow a sequence of instructions (known as a program). These instructions are fetched one at a time from memory and executed. This may require the reading of some information from an input device or from memory, the storing of data in memory, writing to an output device or just some internal manipulation inside the processor. Most processors are capable of holding a very small amount of information internally for such manipulations. Of course, this description is much simplified but will do for now.

SUMMARY

Together these pieces form the basis of the hardware in the typical microcomputer. Having looked at the broad overall structure in very general terms, we can now examine the pieces more closely. Perhaps the best place to start is with the 'information' or data itself and how it is represented within the hardware, since this is fundamental to understanding the rest. So next time, the

subject will be the Binary number system.

- 0000000000 -

***** REVIEWS *****

Reviews of commercially available hardware and software products are presented for the benefit of those who may be considering investing some of their hard-earned dollars. A number of readers have made enquiries regarding the program DUPLITAPE reviewed last issue. It may be obtained from:

ALBION SOFTWARE
LAMAS ROAD
LEYTON
LONDON E10 7QT

If you have recently bought some software or hardware, then send in a review of it and recoup some of your costs. If it's a good product, others will appreciate hearing about it and if it's not, they'll appreciate it even more!

***** ACCEL3 VS. ZBASIC - by R.T. Worley *****

Recently I purchased copies of ACCEL3 and ZBASIC as I wanted to speed up a program that printed scientific formulae (including large integral signs, quotients, matrices etc.) on my EPSON printer.

ZBASIC was the first to arrive, twelve days after I posted off my order. It contained a note apologising for the fact that they couldn't send it airmail as I didn't include $20 postage (I included only $5).$4.10 sure gets fast surface (?) mail. ACCEL3 arrived ten days later. This is quite good service on an overseas order (I ordered from the American distributor in both cases).

ZBASIC came on a disk, together with a ring-binder containing the instructions (about 60 pages) for use. This was despite my asking for a tape in case customs wanted to levy duty, which they didn't despite having opened the package for inspection. ACCEL3 came on tape, together with a plastic bound booklet of instructions.

Copying the programs to my disks was easy. The ZBASIC disk, although it contains no DOS, boots up when put in drive 0 and runs a single drive copy program which copies the ZBASIC files to your own system disk. You do not need two drives. The instructions for loading ACCEL3 to disk are supplied with the program. This involves loading and initialising ACCEL3 from tape (in level II SYSTEM mode), then resetting to boot up the DOS and using the DUMP command of DOS (the instructions tell you which START and END to use). No trouble was experienced with copying either system.

With both compilers you load the compiler from DOS (or SYSTEM mode), enter BASIC and load the program to be compiled (from tape or disk or keyboard). The compiler is activated by special key sequences, and compiles the program in situ. You can then save the compiled program to disk (with the disk versions - ACCEL3 requires TSAVE to save a tape of the compiled program). Compilation is quite fast with each program, though ZBASIC seems a little faster.

With ZBASIC the original BASIC program is left unchanged, so the size of program you can compile is limited. ZBASIC sits slap in the middle of memory (about 6K) leaving equal space for the BASIC and compiled programs. With smaller programs when there is room for the variables above the compiled program you can test run the compiled program, exit it (if you programmed an exit), change the BASIC code, recompile and re-run - all without reloading either BASIC program or compiler. With larger programs there is the facility for relocating the object code to allow room for variables, the constraint being that this will clobber ZBASIC and the BASIC program when you run the relocated object. Since variables often take as much space as programs, the program size limit does not seem much of a consideration.

With ACCEL3 the original BASIC program is overwritten by the compiled program. You must always reload the BASIC program before trying any changes. Since ACCEL3 sits at the top of available memory, taking about 5½K of room, you can compile larger programs if you don't have many variables.

The size of the object code produced by the compilers was about the same in the large program I compiled - in each case it was less than the original BASIC program with remarks included but around 15% bigger than a packed BASIC program. ZBASIC's compiled size includes the run-time module, whereas ACCEL3's compiled size does not. At run-time ACCEL3 requires BASIC to be loaded and an extra run-time module sitting in high memory. In a 48k machine this run-time module sits at and above location 59904 (DEC), and occupies 1½k leaving some relatively useless room above it. Only when compiling in a larger machine for a smaller can you site the run-time module at the top of the smaller machine's memory. At run time ZBASIC does not require BASIC, so in a disk machine you have memory from 5200 (HEX) up free for your program. This gain in

memory compared with the loss in ACCEL3 could be important (however, see later for the possible loss of strings).

ACCEL3 compiles virtually any standard BASIC program without a hitch (actually not all statements are compiled - only a useful subset - but other statements are allowed and passed to the interpreter). Since disk commands are passed to the interpreter, it should handle all DOS commands correctly. If you are planning on compiling existing programs then ACCEL3 is probably the best choice. It may even find errors that you didn't know about (in lines that have never been executed.) The only problem is that you lose a fair slab of memory in a 48K machine. I have some programs that use all but 6K of memory and these just will not fit when compiled.

ZBASIC compiles only a subset of BASIC. This subset contains the integer numeric variables and string variables, and integer and string functions. Higher precision numeric variables and arithmetic are available (but are a little awkward to use at first) and, as they are coded as strings, they consume more memory. Access to scientific functions like SIN, LOG must be done through machine language calls to ROM - this is awkward and errors can crash the program. There are extra functions available - PEEK and POKE can have words as arguments (PEEKW(16526) returns the two-byte word stored at 16526,7), AUTO is a tone generation function which can send a tone of variable frequency and duration to the cassette port. Access to machine language programs, and embedding of machine language routines in the program, is very easy. ZBASIC also includes some machine-language-like commands - there are commands emulating LDIR, LDDR, CPIR etc.

Although ZBASIC restricts the BASIC program language, it is adequate for many uses except where scientific functions or extremely small or large numbers are used (its high precision arithmetic is binary coded decimal and does not use exponential notation). It has its own limited file handling system, handling sequential and random access files, but the latter must have record length 256. In many cases the ease of access to machine language compensates for some shortcomings. However, there is one important point. Because of the way they are stored, strings are allocated a fixed maximum length. All simple string variables are allocated a length (L1 say), and all array string variables are allocated a length (L2 say). These values can be specified when loading ZBASIC. However, if you want one array ST of dimension 10 to have long strings up to length 255, then you set L2=255 and the program will allocate 2550 bytes for this array. If you want another array SU of dimension 100 to have strings of length at most 5 then you must still allocate L2=255 (because of ST). So you use 25500 bytes for the array SU and not 500, unless you cut the maximum length of ST. This chews up memory rapidly unless you program your way round it. Another consequence is that strings cannot contain nulls since a null is the terminator. Now you may rightly say "who on earth uses nulls in strings?". Unfortunately, I have to send a null to my printer to terminate control sequences. The only way to do this is to use a machine language call (which is easy in ZBASIC, so this compensates a little).

Debugging of compiled programs is a little more difficult. With ACCEL3 you can (BREAK) the run. The variable table is accessible, so you can type PRINT A to print the value of the variable A, etc. By examining the variables it may be possible to determine the cause of the error. Once you (BREAK) the run you may not be able to CONTinue the run. Note that any error messages that appear when the program crashes are only partially correct. Line number references may be wrong. With ZBASIC debugging is more difficult. The variable table is not accessible, and the compiled program runs so quickly that frequent PRINT statements cause the screen to scroll rapidly. Much more care should be taken in writing the program initially. Note that ZBASIC does not seem to check anything. It will quite happily set AR(50000) = 12 when you have set DIM AR(10). This may clobber an element in another array. Of course, this could corrupt the program itself and cause a crash. Arithmetic overflow (50000+50000 = -31072) likewise does not cause an error message. The advantage of less checking is, of course, higher speed.

Since I bought the compilers to compile a specific program, the results for that particular program may be of interest. The uncompiled program initially took 18 minutes to print a page of mathematics. It took me less than ½ hour to have a working version with ACCEL3 (fixing a few syntax errors that hadn't been met by the interpreter). This took 6 minutes to print a page, but made clear that "garbage collection" was a problem. The program printed 4 lines quite quickly, then paused a while, then printed 4 more lines, etc. I revamped the program so the BASIC program took 15 minutes a page and the compiled version took 3 minutes. Garbage collection went down to once a page. The entire job took one evening.

When I tried ZBASIC I first found that I had to pack the BASIC source by deleting all REMarks, to fit below the compiler. I then deleted all syntax errors (ZBASIC requires slightly different syntax in some cases). Since ZBASIC's syntax check is rapid, this was not too slow a job. Then I discovered that I had 10 long strings and 130 short ones, and this doesn't work at all (as mentioned above) as I ran out of room. I went to bed and read the manual carefully. I discovered a way around the problem using some pseudo machine code. Next evening I patched this into the program, and got it working - partially. Whenever I tried certain printer codes, the printer started printing hieroglyphics. That evening I went to bed with the manual and did some deep thinking. After a while I sprang up, raced to the computer, and compiled and ran the one-liner PRINT LEN(STRING$(30,0)). YES!! Answer zero and not 30 - nulls couldn't be embedded in strings. The next two evenings were spent working out how to neatly program my way round that. On the fifth night after receiving ZBASIC I finally had my program working.

Was it worth the effort? The print speed was less than 1 minute a page. Since the printer itself takes over half that, I believe the ZBASIC compiled program operates 5 times faster than the ACCEL3 one.

Another advantage of the ZBASIC compiled program is that it loads directly from DOS as a command file, whereas the ACCEL3 version requires loading the run-time module from DOS, entering BASIC specifying memory size and number of files, loading the compiled program, initialising the run-time module with a SYSTEM call, and finally RUNning the program. I would recommend ZBASIC for its speed and ease of use of the finished program, but warn of restrictions, in the available language. If you can program in machine language it would be a help.

ACCEL3 (copyright 1982 Southern Software)
Purchased from ALGORIX Software, Box 11721, San Francisco, CA 94101, USA. Cost approx. $US100.

ZBASIC2.2 (copyright 1982 Simutek Computer Products and Andrew Gariepy)
Purchased from SIMUTEK Computer Products Inc., 4897 E. Speedway Blvd., Tucson, Arizona 85712, U.S.A. Cost approx. $US95.

- 0000000000 -

***** LNW-80 MK II - A HARDWARE REVIEW *****

by John L. Reichelt

The system: LNW-80 mark II computer
Zenith hi-res green screen monitor
Twin MPI B52 disk drives
Dosplus 3.4 and LNWCPM 1.0 DOSes
LNWBasic extended Basic interpreter
Tandy Daisy Wheel Printer II.

My first computer (still hard at work) was a 1978 version of the model I TRS-80 using cassette and ESF so, in reviewing the LNW-80 mark II, I will try not to confuse the issue with my enthusiasm for my new MPI disk drives and Dosplus 3.4.

I purchased the LNW-80 first and foremost because I wanted a new computer with complete (i.e. 100%) compatibility with my Model I TRS-80. A lot of new features is fine but not if I have to throw away old programs and give up my limited but hard-won knowledge. After two months of use, I am delighted with the degree of Model I compatibility achieved by the LNW-80. BASIC programs, machine-language programs and hybrid (BASIC plus machine language) programs are all running fine without modification. I have had some fun successfully converting certain tape-based games and tape-based Scripsit files to disk but that is not unique to the LNW-80. The one definite case of a problem specific to the LNW-80 has occurred with a machine language game, Penetrator. The LNW-80 runs this game only in reverse video, which is unsatisfactory. Reverse video on the LNW-80 is performed by the command OUT 254,1. From my attempts to run Penetrator, I feel that the reverse video is being invoked when the game is generating sound via the cassette port (255). A number of other programs generate sound via the cassette port without causing this problem, however.

The LNW-80 is advertised as running at 4 MHz which, although it sounds nice, worried me as a potential source of incompatibility with Model I TRS-80 programs (I tend to worry a lot). In fact, due to the provision on the keyboard of a lockable key to switch between 1.77 and 4 MHz, no problems arise. Cassette and ESF wafers with programs written at 1.77 MHz have to be read with the High/Low processor speed key locked down. However, programs can be stored and retrieved at 4 MHz using cassette and ESF. Whenever the disks are in operation, the processor speed becomes 1.77 MHz automatically. When running programs, you can switch back and forth between 1.77 and 4 MHz to speed up the slow parts or slow down the difficult parts of the program. Many games have to be played at 1.77 MHz to be feasible. Scripsit at 4 MHz is very pleasant indeed.

I have read reviews of the LNW-80 Mark I which commented favourably on all aspects except the keyboard (keybounce and general touch) and the manual (lack of). After my previous broad experience of one 1978 Model I TRS-80, I find the keyboard of the LNW-80 Mark II a delight to use. Under some conditions - generally when my tired fingers start dragging - the auto-key-repeat and 4 MHz leads to several letters per key press. This can be alleviated by waking up or pressing the 1.77 MHz key. The manual for the LNW-80 Mark II exists but it is a bare-bones minimum. However, the only thing I have been specifically prevented from doing through lack of information in the manual is using the user-definable keys. The keys are labelled f1 and f2 and currently produce down arrow, right arrow, the yen sign (shift f2) and the mod sign (shift f1). The manual says these keys are user definable but doesn't say how. I can change them using the Extended Basic program so I am sure there is some way to define these keys in simple TRS-80 mode.

In summary, the LNW-80 Mark II is a superb replacement for the TRS-80 Model I. What else can it do? Well, firstly, it has all this extra RAM lying about. Some of this is used to set up

CP/M mode. CP/M is supplied as part of the package and you can set up 48K and 64K CP/M modes. Not yet having any CP/M programs, all I have done with this is to set it up and see that the CP/M operating system is, in fact, there...and it is. Of more immediate use to me is the 32K of extra RAM which can be swapped with the top 32K (8000H to FFFFH) when running in TRS-80 mode. This is done using the BASIC command OUT 31,64 or the assembly language equivalent, OUT (1FH),40H; the information stored in the original 32K memory block can later be recovered using OUT 31, 32 or OUT(1FH),20H. All of this works fine but, if used with BASIC programs, you have to remember to protect memory at 7FFFH ; otherwise the stack and string space area are abruptly removed!

A further 16K of extra RAM is involved in another major innovation of the LNW-80, high resolution graphics. Implementing the 480x192 B/W hi-res is very easy using the Extended Basic supplied. Using the hi-res in simple TRS-80 mode without the Extended Basic is more complex but, fortunately, the graphics section of the manual is more detailed than the rest of the sections, and a machine language routine for hi-res SET, RESET and POINT commands is supplied. The hi-res is achieved without loss of compatibility with the TRS-80 Model I by temporarily banking in the required 16K of graphics RAM into the first 16K of memory addresses, thereby temporarily banking out the ROM. This system also operates in colour but I have not tested this at all.

Over the last month, I have found the LNW-80 Mark II to be a highly reliable and stable computer system. The only time I have lost data was in a complete blackout. I find that I can even turn on the Daisy Wheel printer while the computer is running, an insult my TRS-80 Model I goes to pieces over. However, when I first received the computer, I had considerable difficulty with unstable video and computer resets. An investigation showed that this was due to the use of a low voltage supply. The LNW-80 specifications call for 120 volts. Tests on my machine showed that the video image starts to shimmer at 105 volts and the computer becomes unstable below 100 volts. This means that you can use a 100 volt transformer if you work in an industrial zone getting 260 volts down the line (which is where I am now), but if you get 240 volts or less, you would be better off with a 115 or 120 volt transformer.

The LNW-80 Mark II has so far satisfied most of my desires for advanced features and saved me from the difficult decision of which incompatible new computer system is worth the loss of my present programs and knowledge. Best of all, my new computer is still supported by MICRO-80, whose staff expended considerable effort to deliver my machine before Christmas...for which I am very grateful.

(The LNW Research Corporation has improved the new LNW80 Mark II computer over the model delivered in 1982. As Mr. Reichelt has ordered a second LNW 80 II since writing this review, these improvements should be of interest.

Fortunately, these changes have not introduced any incompatibility whatsoever with the previous model.

Many of the improvements are related to the video display. The resolution of the low-res colour graphics mode has been increased from 128x192 pixels to 160x192 pixels, still with 8 colours. In addition, control of background colour has been added and the background colour can be set under software control from LNWBASIC.

The new model allows colour programs to be run without a colour monitor by providing 8 levels of video intensity at the monochrome video output. However, this single colour display is limited to 128x192 pixel resolution in low-res mode. Hardware inverse video by character has also been added.

The graphics RAM now has provision for four pages of screen display. The six 16K RAM chips can be replaced with 64K RAM chips to give four pages of graphics memory. This means that you can display one page while modifying another and then switch in the new page for a cleaner visual effect and fast animated graphics.

A small loudspeaker is now built-in with a volume control on the back panel. As well as being compatible with most arcade-style games, the sound output is supported by the SOUND command in LNWBASIC.

In addition, a number of small refinements have been made to some already existing features. The RS-232 port now provides software programmable baud rates and configuration. The RGB video connection has been improved to provide easier interfacing to a greater range of colour monitors. The cable connectors for the floppy disk and printer port have been reversed so that the cables now lay down for easier connection. The Level II BASIC interpreter has been enhanced and the power supply improved for reduced internal temperatures and greater reliability. Minor changes have been made to the video display to remove incompatibility with some TRS-80 programs and games.

LNW have managed to improve the Mark II without increasing its cost. Unfortunately, the devaluation of the Australian dollar has meant that the price here will definitely be higher. I look forward to hearing more about the LNW computer and its capabilities in the near future. - Ed.)

- 0000000000 -

***** MICROSOFT LEVEL III BASIC *****

A Review by A.G. Hudson, M.A.

LEVEL III BASIC by Microsoft is one of the many approaches to giving the user a bit more in the way of BASIC language commands, together with a few other conveniences which do not come when you buy your machine with its 12K of LEVEL II in ROM. The LEVEL III facilities come in software in the form of a tape that loads under the SYSTEM Command, and therefore they use up a slice of the available RAM, 5.5K in fact. It is possible to store LEVEL III on disk, but once loaded into the machine, you will no longer have access to disk storage commands. A number of the extras in LEVEL III are among those that come with Disk BASIC, these include INSTR, DEFUSR, LINE INPUT and long error messages.

However, there are other facilities which you will not get from disks and which may be just what you want to help your programming. It is as an aid to programming that I have found LEVEL III most useful.

To start with, there are 26 abbreviated entries obtained by shifting each letter of the alphabet. For instance, shift H = INKEY$ which saves quite a bit of typing if you have a lot of INKEY$ in your program. However, you do not have to take the abbreviated entries as given. Using the command LSET you can create your own, and they can be any length up to 15 characters. So, if you have a program to type in where there is frequent use of a sub-routine at line 9000 try LSET A = "GOSUB 9000" and every time after that shift A will do the job for you. I reckon there is a very useful saving of time by working in LEVEL III even when working on LEVEL II programs.

In LEVEL III SAVE and LOAD work to tape and are less sensitive to changes in volume level than CSAVE and CLOAD. It is helpful to be able to abort either with the break key.

The NAME command allows re-numbering of all or part of the program starting at a specified number. The new numbers can start at any legitimate number and increment by any chosen amount. Moving the later part of the program onto a higher series of numbers can be a useful way of creating space for additional lines in the middle. However, you cannot use NAME to alter the sequence of lines.

Quite different are the LEVEL III Graphics. These I have not found anywhere else. Just as in LEVEL II you can use the graphics symbols, CHR$ (129) to CHR$ (191), but placing them not at numbered spaces, but at spaces identified by the co-ordinates X, Y. X is between 0 and 63 and Y between 0 and 15, using the LINE statement. Alternatively, LINE with SET or RESET works on individual PIXELS as does SET and RESET in LEVEL II. By giving two sets of co-ordinates the line instruction will draw the line joining the two (or erase it if using RESET). Adding "B" to the LINE statement will draw the outline rectangle of which the co-ordinates used are diagonally opposite corners and adding a further F fills it in solid. This certainly saves a lot of laborious programming compared with LEVEL II. GET will store in an array the contents of a specified section of the screen. PUT will place the array back on the screen. By specifying different co-ordinates for the section of the screen to be used, the original graphics can be moved from one place to another. Additionally, if the graphics mode is used when the array is PUT on the screen it can be put in SET or RESET mode. It can also be combined with existing display by AND, OR or XOR statements, which offers many intriguing possibilities.

For those who have an expansion interface, LEVEL III also provides a Real Time Clock and Calendar. The date has to be set American fashion - Month, Day, Year, two figures for each and leading "Ø" must be included. Personally I do not go for the clock facility. If you forget to turn it off before carrying out tape input or output (except SAVE and LOAD which turn the clock off and on again automatically) then it is all too likely that the input or output will be corrupted. Given the time it takes to SAVE or LOAD or do any other tape operation on the TRS-80, the clock is not going to be accurate for very long.

The copyright date for Microsoft LEVEL III is 1979 and it has now been available (in England) for three years. I have yet to see any program published or advertised requiring LEVEL III to run (nor, for that matter, have I seen any based on the other extended BASICS which are on the market).

It may be lack of enterprise; it may be that others use it as I do to speed up work on LEVEL II programs. Those who want fast graphics can do it by POKEing direct into the video display addresses and there will always be the exponents of machine code. Most probably it is because only those having LEVEL III will be interested in LEVEL III programs, and nobody knows how many of us there are. It is also natural to move into DISKBASIC and one cannot pursue every interest at the same time. LEVEL III does not mix with lower case, so how will it go with model 3? On the face of it, the higher baud rates speeding up cassette input and output could make tape a more attractive proposition. LEVEL III might then come into its own.

The booklet Program Instructions for Microsoft LEVEL III BASIC is excellent, covering all the features in a way which is easy to follow. One can be using LEVEL III within five minutes of getting it, though it will take some time to learn to exploit all of its features fully.

In conclusion, I would say that if you are more interested in writing your own programs in BASIC than in working with large files, or if you want to do some really enterprising experimenting in graphics, the LEVEL III is a worthwhile buy. The other facilities also tend to come in other packages, and if it is one or more of these you really want, then look at the others to see

if one of them is not a better proposition for you.

- 0000000000 -

***** DICK SMITH'S WORP-1 WORD PROCESSING PROGRAM (16K CASSETTE VERSION) *****

A Review by M.L. Rule.

It was Edgar Allen Poe who, as I recall, was much disturbed once upon a midnight clear by a strange tap, tap, tapping noise. We had a similar problem at our place until recently but, instead of Poe's spooky raven, our problem was me transferring data manually from the VDU of my computer system to paper via a rather ancient typewriter.

I had a problem at work which involved a fair bit of polynomial regression and, as our EDP manager does not recognise as a computer anything that does not have twin 32-bit CPU's and a zillion megabytes of mass storage, I quietly took the data home and produced the required results in a couple of days by crunching it through the old System 80.

This made me feel good, impressed my management no end and demonstrated the usefulness of small systems in dedicated applications in a large organisation. There were, however, two unexpected drawbacks - the EDP manager doesn't talk to me anymore and the analysis showed that we had to do a whole lot more of this sort of analysis...and guesswho got stuck with the job...

I am beginning to suspect that Dick Smith has some sort of information sharing deal with my bank manager because, just as my special savings account towards an extra 32K of memory and perhaps a disk or two crawled past $500, he started to advertise a line printer for $495. Not only did he advertise this little marvel but he actually had some in stock and, as my eyes were becoming permanently red-rimmed from my midnight labours, I reluctantly kissed goodbye to the 32K and the disks and bought the printer.

This solved the red-rimmed eyes problem and further impressed my management with nice, professional looking hard copy of the data, and I began to look for further uses for the printer. I have the sort of job where I dispense pearls of wisdom both verbally and in writing and, as I plaintively tell anyone who will listen, I tend to talk all day and write all night. An obvious application therefore was to make the old typewriter totally redundant and get into word processing...and it just so happened that Dick had a new, U-beaut word processing program available.

My first impressions of WORP-1 were favourable. Listening to the tape showed that the recording was crisp and clean and the program loaded perfectly. The program comes with 30 pages of bound documentation which includes an overview of word processing, an excellent description of the system and its features, instructions on how to change the printer and program parameters for different column and memory sizes, and a complete listing of the cassette and disk versions of the program.

The program was obviously written for plutocrats with lower case mods and daisy wheel printers as the print out from my first attempts at using it was all lower case. A quick look at the program showed that the cunning author had converted the computer keyboard to normal typewriter mode by fiddling with the decimal codes for the ASCII characters. As I don't have a lower case mode (yet) and as the printer has those funny descenders, I changed the program to convert the decimal values on which it works back to those for upper case. That solved the lower case problem.

I was also ecstatic about the automatic text wrap-around feature. After years of running off the end of the page on the old Remington (the warning bell gave up the ghost years ago), it was an absolute delight to just be able to continue typing, knowing that the machine was taking care of all of the housekeeping for me. My first shock came when, whilst in full creative flight, I looked up to see that the last four or five words that I had typed were not on the screen (I am a two-finger, "pick and peck" typist) and that the flashing cursor had disappeared. After about thirty seconds it re-appeared and everything was back to normal. On reading the documentation, which I should have done in the first place, I found that this was caused by the Level 2 BASIC string handling routines which periodically shut everything down to do some internal garbage collection. It is more of an annoyance than a problem as it invariably happens when I am about to get down some particularly telling point and, by the time the system has returned to life, I have forgotten the point that I was about to make.

Our first major disaster reduced my 14 years old son to tears of frustration when, after many hours of laborious composition of a letter to his cousins in Melbourne, the program jumped back to the menu and nothing that he or I could do would recover the text. The program was obviously written on a 32K system because, on delving into it, I found that the array that holds the word count had been dimensioned down to fit a 16K system, but that the warning routine that alerts you to the fact that the document is full still had the 32K figure. Once the word count dimension has been exceeded, a beyond subscript error occurs and the error handling routine in the program returns you to the menu. If you buy the program, change the figure in line 250 from '975' to '700'.

MICRO-80 PRODUCTS CATALOGUE

This catalogue contains a selection from the wide range of peripherals, interfaces, computers and software carried by MICRO-80 for your computer. If you don't see the item you want, contact us, we probably have it anyway!

MICRO-80 has been supplying customers throughout Australia and the Pacific region by mail-order for 2½ years. Our customers find this a simple and efficient way to do business. You may place your order by telephone or by mailing the order form from any issue of MICRO-80 magazine. Generally, it takes about one week from receipt of order until despatch. You should allow 2-3 days for your letter to reach us and 7-10 days for the parcel to reach you, making a total turnaround time of 2½-3 weeks.

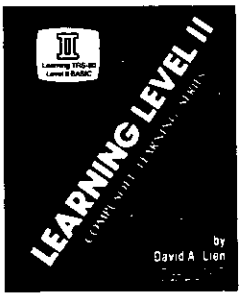
WARRANTY AND SERVICE

All hardware products carry a 90 day parts and labour warranty either from the manufacturer/distributor or from MICRO-80 Pty Ltd. In many cases, warranty servicing can be arranged in your own city, otherwise goods will be repaired by our own team of technicians in our Adelaide workshops.

TRADE-INS AND TERMS

MICRO-80 can accept your existing equipment as a trade-in on new equipment. We can also arrange consumer mortgage financing or leasing on larger hardware purchases. Contact us for details.

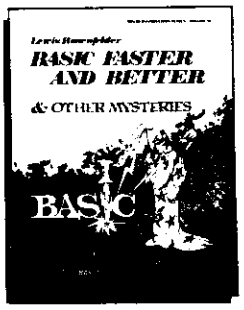
BOOKS



LEARNING LEVEL II
by David A. Lien

Written by the author of the Level I Users Manual, *Learning Level II* covers all Level II BASIC beyond Level I, plus much more. It shows you how to use the Editor, explains what the many error messages are really saying, and leads you through conversions of Level I programs to Level II.

Dual cassettes, printers, the Expansion Interface with clock and other features are explained in the same easy-to-learn style that made the Level I Manual famous. *Learning Level II* is an invaluable supplement to the TRS-80 Level II and System 80 manuals and is now only $7.95 (plus $1.20 p&p).



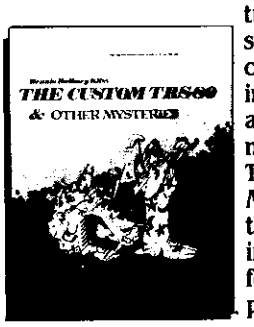
BASIC FASTER AND BETTER AND OTHER MYSTERIES
by Lewis Rosenfelder

Basic is not nearly as slow as most programmers think. *Basic Faster and Better* shows you how to super charge your BASIC with almost 300 pages of fast, functions and subroutines.

You won't find any trivial poorly designed "check-book balancing" programs in this book — it's packed with *useful* programs.

Tutorial for the beginner, instructive for the advanced, and invaluable for the professional, this book doesn't just talk . . . it shows how! Basic Faster and Better is $32.50 (plus $1.20 p&p).

BOOKS

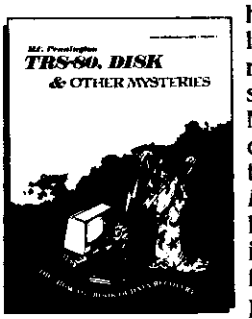


THE CUSTOM TRS-80 AND OTHER MYSTERIES
by Dennis Bathory Kitsz

Ever wanted to do things to your TRS-80 that Radio Shack said couldn't be done? How about reverse video, high resolution graphics, and audible keystrokes?

Now enough? How about turning an 8-track into a mass storage device, making music, controlling a synthesiser, individual reverse characters, and a real-time clock just to name a few?

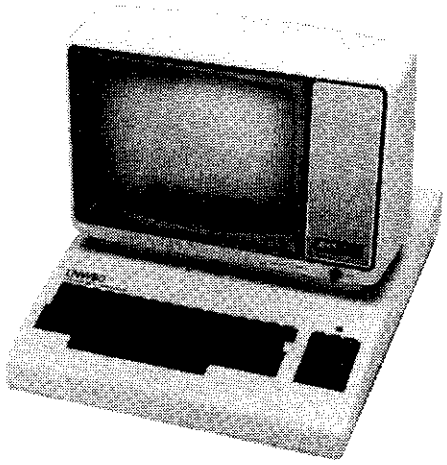
The *Custom TRS-80 and Other Mysteries* is packed with more than 290 pages of practical information and can be yours for only $32.50 (plus $1.20 p&p).



TRS-80 DISK AND OTHER MYSTERIES
by H.C. Pennington

TRS-80 Disk and Other Mysteries is the definitive fix-it book for disk users. More than 130 pages of easy to read, entertaining and immensely useful information. Find out how to recover disk files, the layout of information on disks, memory maps, problem solutions . . . the list goes on! Many readers have saved days of work by recreating disk files that were unreadable. *TRS-80 Disk and Other Mysteries,* which has received favorable reviews in several magazines, is yours for only $27.00 (plus $1.20 p&p).

THE LNW80 MkII MICROCOMPUTER



Manufactured in America by LNW Research Corporation, the LNW80 II has the following outstanding features:

- Completely software and hardware COMPATIBLE with the TRS-80 Model 1.
- HIGH RESOLUTION COLOUR GRAPHICS — 4 MODES:
  - B/W LO-RES 128 x 48
  - B/W HI-RES 480 x 192
  - COLOUR LO-RES 128 x 192 in 8 COLOURS
  - COLOUR HI-RES 480 x 192 in 8 COLOURS
- CP/M Disk Operating System.
- Single and Double Density Disk Operation.
- Supports 5¼ inch or 8 inch Floppy Disk Drives.
- 48K RAM in TRS-80 mode plus 16K High Resolution graphics RAM.
- 64K RAM in CP/M mode plus 32K Banked in, usable in BASIC, plus the 16K High Resolution Graphics RAM.
- 4 MHz Z80A microprocessor — over twice the operating speed of the Model 1.
- HI-RES COLOUR (R-G-B) and B&W video outputs.
- 3 screen display modes:
  - 64 characters x 16 lines
  - 80 characters x 16 lines
  - 80 characters x 24 lines
- SOFTWARE SUPPORT

Apart from being able to run all TRS-80 Model 1 software and all CP/M software, there is also an extended BASIC interpreter available for the LNW80 II using most of the same commands as the TRS-80 Colour Computer but with full LNW Graphics Resolution, SET, RESET, POINT, LINE and CIRCLE as well as special commands to generate sound effects and tones. TRS-80 Colour Computer BASIC programs can be transferred to the LNW with only minor changes.

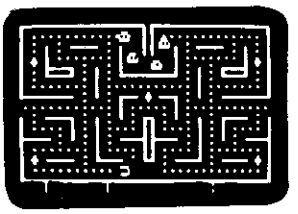
Prices include Sales Tax and are subject to change without notice. Prices are FOB Adelaide. Add $20 road freight anywhere in Australia.
All equipment carries MICRO-80's Australia-wide 90-day warranty covering parts and labour.

The LNW80 II is the ideal computer for the serious hobbyist or businessman who is seeking a higher performance, more reliable computer to replace his TRS-80 Model 1 without sacrificing his investment in software or his programming experience. The LNW80 II uses standard Tandy or Tandy compatible disk drives. If you already have a disk TRS-80 system you may continue to use your existing disk drives on the LNW80 II.

LNW80 II Computer — complete except for disk drives and monitor Includes:

— CP/M Disk Operating System Dosplus 3.4 Double Density Disk Operating System
— LNW Extended Colour Basic Interpreter ..... **$2750** INC.S.T.

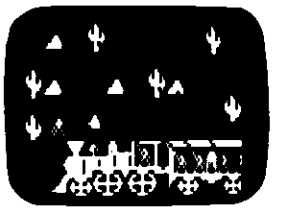
HI-RES Green Phosphor Monitor ..... **$265** INC.S.T.
Super HI-RES Hitachi RGB Colour Monitor ..... **$1250** INC.S.T.
Two Singlesided 40 Track Double Density Disk in cabinet with power supply and cable ..... **$825** INC.S.T.



SCARFMAN

This incredibly popular game craze now runs on your TRS-80! It's eat or be eaten. You run Scarfman around the maze, gobbling up everything in your path. Try to eat it all before nasty monsters devour you. Excellent high speed machine language action game from the Cornsoft Group. With sound.

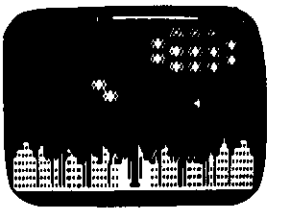
Price: $17.95



THE WILD WEST

It's up to you to keep the West beautiful with Outlaws and renegade Indians on all sides. Even the train has been captured by Outlaws with all the payroll on board. Can you clean up the Wild West?

Price: $26.50



SPACE ATTACK

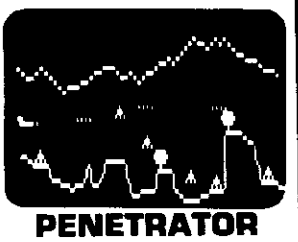
Steady your nerves, keep a sharp lookout, and prepare for battle to save your city. Fiendish aliens are all around, and if they destroy the city you lose.

Price: $26.50

STRIKE FORCE

As the primary defender of a world of cities under deadly alien attack, your weaponry is the latest: rapid fire missiles, long range radar, and incendiary "star shells." Your force field can absorb only a limited number of impacts. A complex game of strategy, skill and reflexes from Melbourne House.

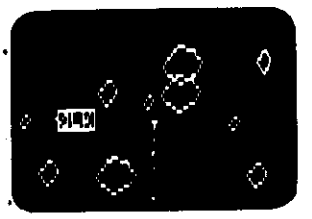
Price: $26.50



PENETRATOR

Soar swiftly over jagged landscape, swooping high and low to avoid obstacles and enemy missiles attacks. With miles of wild terrain and tunnels to penetrate, you're well armed with bombs and multiple forward missile capability. From Melbourne House. Features sound, trainer mode and customizing program.

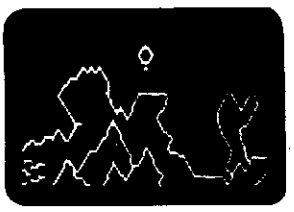
Price: $36.50



SUPER NOVA

Asteroids float ominously around the screen. You must destroy the asteroids before they destroy you! (Big asteroids break into little ones). Your ship will respond to thrust, rotate, hyperspace and fire. Watch out for that saucer with the laser! As reviewed in May 1981 Byte Magazine.

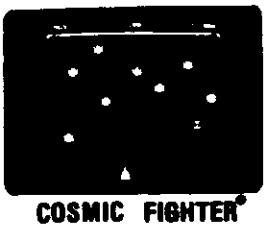
Price: $26.50



LUNAR LANDER

As a vast panoramic moonscape scrolls by, select one of many landing sights. The more perilous the spot, the more points scored -- if you land safely. You control LEM main engines and side thrusters. One of the best uses of TRS-80 graphics we have ever seen. From Adventure International. With sound.

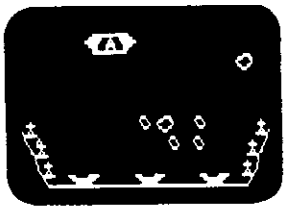
Price: $26.50



COSMIC FIGHTER

Your ship comes out of hyperspace under a convoy of aliens. You destroy every one. But another set appears. These seem more intelligent. You eliminate them, too. Your fuel supply is diminishing. You must destroy two more sets before you can dock. The space station is now on your scanner... With sound!

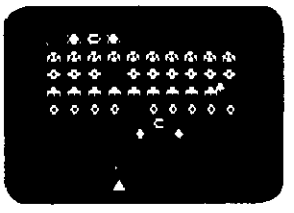
Price: $26.50



METEOR MISSION II

As you look down on your view, astronauts cry out for rescue. You must maneuver through the asteroids and meteors. (Can you get back to the space station?) Fire lasers to destroy the asteroids, but watch out, there could be an alien Flagship lurking. Includes sound effects!

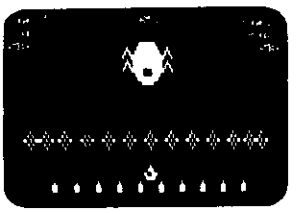
Price: $26.50



GALAXY INVASION

The sound of the klaxon is calling you! Invaders have been spotted warping toward Earth. You shift right and left as you fire your lasers. A few break formation and fly straight at you! You place your finger on the fire button knowing that this shot must connect! With sound effects!

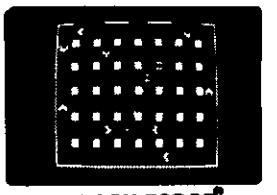
Price: $26.50



DEFENSE COMMAND

The invaders are back! Alone, you defend the all important nuclear fuel canisters from the repeated attacks of thieving aliens, repeatedly. An alien passes your guard, snatches a canister and flys straight off. Quick! You have one last chance to blast him from the sky! With sound and voice.

Price: $26.50



ATTACK FORCE

As your ship appears on the bottom of the maze, eight alien ships appear on the top, all traveling directly at you! You move toward them and fire missiles. But the more aliens you destroy, the faster the remaining ones become. If you get too good you must endure the "Flagship". With sound effects!

Price: $26.50

FOR YOUR ENTERTAINMENT

MICRO-80 now offers you the widest range possible in entertainment software. These programs are supplied on cassette for the Level II/16K TRS-80 Model I/III (except as noted). They are also suitable for the System 80 but sound may not be available unless a hardware modification has been fitted to reverse the roles of recorders #1 and #2. *Order yours now while stocks last!*

DEFENCE PENETRATOR $20.95
DEFENCE PENETRATOR is based on one of the most popular arcade favourites of all time with smooth graphics and sound effects. With realistic scrolling planetscape it's the best game yet.

DEVIL'S TOWER $25.95
Aliens move in waves of 5 attackers with their robot scouts attacking you from the mountain, their war machines and their protector ships putting up force fields to protect them. Only your skill and fast reflexes can save the plant.

BATLE STATION $21.50
The aim of the game is to defend your space station against the attack of four alien space ships.

MORGOTH $20.95
Morgoth is a unique action packed adventure allowing you to wander through the enchanted dominion of Morgoth and collect the lost treasures of KAZARD KALLAHAN. But Beware! You must escape before the satanic Morgoth is aroused and seeks yea!

KILLER BEETLES $21.50
The aim of the game is to dig traps. When a beetle falls in you must fill it in to bury them, before they can catch you.

STAR CRESTA $20.95
Star Cresta takes you beyond the limits of your computer and into the Cosmic void itself! Beware! Iron clad concentration and lightning relfexes are required to destroy the evil empress.

JUNGLE RAIDERS $21.50
The aim of the game is to defend your four bases from the marauding Jungle Raiders. Your skill all the Jungle Raiders and they try to hit you with their spears or drag off all four of your bases.

ALIEN TAXI $28.50
Your goal is to pick up and deliver passengers to an underground resort hotel. There is a fare at each of the 12 taxi stands on the first level and 12 more on the second level.

KILLER GORILLA $21.50
Four completely different frames. Each one offering a different challenge, makes this one of the most complex and stimulating games ever written for a TRS-80. The game keeps track of the top ten scores along with a six character name for each score.

JUNGLE BOY $21.50
The ultimate challenge! Are your reflexes fast enough to swing Jungle Boy from vine to vine? Can you swing through the jungle? Can you swim by the alligators? These are just some of the things you will find very challenging in Jungle Boy.

STELLAR WARP $20.95
Animation with superior fighter craft brings you an even greater challenge. As your computer advances your level, the aliens become more dangerous and the harder it is to stay alive!

HOPPY $21.50
The aim of the game is to get your frogs across the busy highway without being squashed and then across the river by means of floating logs and turtles.

PANIK $28.50
Your mission is to rid the galaxy of the Mzors forever. Mzors are half animal and half machine. Their leaders are very difficult to destroy and are capable of creating more warriors at will. Your weapons are your energy pistol, short range transporter pack and your courage.

INSECT FRENZY $21.50
The aim is to stop the centipede from getting you, all the time keeping an eye out for the giant spider.

ALIEN CRESTA $21.50
The aim is to defend your ship from numerous attacks from an assortment of aliens. If you get hit three times, it's all over.

DESERT PERIL $28.50
The Zagons have mined the desert and have put killer satellites, drone bomber balloons, and flying dragons along the whole trail. The future of your planet's race depends on your skill and daring.

RALLY RACER $20.95
Drive through an action packed maze and try to hit all the flags before Morgan the Mad motorist or Crazy Harry and his killer hoodlums catch you!

NOTE:
As the prices of imported software may vary, these prices are valid for current stock only and prices are subject to change without notice.

BUY YOUR MODEL 3 FROM MICRO-80 AND SAVE $00's

MICRO-80 fits reliable MPI disk drives to the TRS-80 Model 3 to give system capacities and capabilities far in excess of those available elsewhere. All our conversions utilise low dissipation, switching-mode supplies to avoid screen jitter and overheating. The disk controller boards used incorporate special compensation circuitary for 80 track disk drives and may also be used to run 8 inch disk drives with an appropriate cable and DOS.

MODEL 340 **$3130**

2 40 TRACK SINGLE-HEAD DISK DRIVES GIVING
350K FORMATTED STORAGE, 48K RAM

MODEL 340+ **$3350**

2 40 TRACK DUAL-HEAD DRIVES GIVING
700K FORMATTED STORAGE, 48K RAM

MODEL 500 — 5+ MEGABYTE MODEL 3 **$5895**

1 40 TRACK DUAL-HEAD DRIVE GIVING 350K
OF FLOPPY DISK STORAGE FOR TRANSFERRING
PROGRAMS AND BACKUP, 48K RAM, EXTERNAL
5 MEGABYTE WINCHESTER SUB-SYSTEM,
DOSPLUS 4.0 DISK OPERATING SYSTEM
The MODEL 500 offers the high speed, mass storage capacity and reliability of a Winchester drive for thousands of dollars less than you would pay for any comparable system. Model 500 is a serious business computer able to tackle the most demanding tasks.

WINCHESTER DISK DRIVE SUB-SYSTEM **5MByte $2995**
10MByte $3750

This Winchester Disk Drive sub-system provides either 5 or 10 Megabyte of reliable, high speed storage. It connects to any standard Model 3 equipped with one or more floppy disk drives and does not void the Tandy warranty. Complete with DOSPLUS 4.0 Disk Operating system.

Prices include Sales Tax and are subject to change without notice. Prices are FOB Adelaide. Add $20 road freight anywhere in Australia. All computers and peripherals carry MICRO-80's Australia-wide. 90-day warranty covering parts and labour.

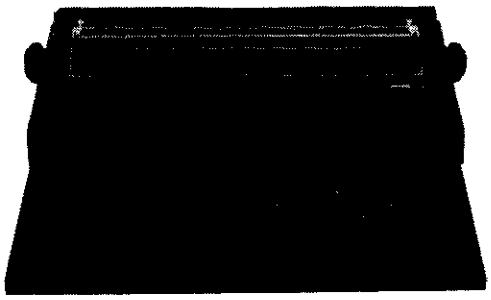
Daisy Wheel Printers/Typewriters

OLIVETTI PRAXIS 35

$895

plus $10 road freight anywhere in Australia

OLIVETTI ET-121



$1500

plus $20 road freight anywhere in Australia

MICRO-80 has converted these OLIVETTI typewriters to work with the TRS-80, SYSTEM 80 or any other microcomputer with a Centronics parallel port. Now you can have the best of both worlds — an attractive, modern, correcting electronic typewriter which doubles as a correspondence quality Daisy Wheel printer when used with your microcomputer.

The **PRAXIS** is a portable typewriter, designed for private and light commercial use with an average print speed of 12 c.p.s.

The **ET-121** is a large typewriter intended for heavier duty and features a print speed of up to 17 c.p.s.

Centronics printer cable to suit TRS-80 or SYSTEM 80 . . . . . $39

MICRO-80 is an A-Grade Olivetti distributor and has been producing printer conversions for Olivetti daisy wheel typewriters for several years. Write or call for full details.

16K Memory Upgrade Kit

$30

plus $2.00 p. & p.

Large volume means we can buy better and can pass the savings on to you. There are our proven, prime, branded 200 nanosecond chips, guaranteed for 12 months. A pair of DIP shunts is also required to upgrade CPU memory in the TRS-80 — these cost an additional $4.00. All kits come complete with full, step-by-step instructions which include labelled photographs. No soldering is required. You do not have to be an experienced electronic technician to install them.

Lower Case Modification

$49

plus $2.00 p. & p.

The MICRO-80 modification features true below-the-line descenders, a block cursor and symbols for the 4 playing-card suits. Each kit comes with comprehensive fitting instructions and two universal lower-case driver routines on cassette to enable you to display lower case. These routines are self-relocating, self-protecting and will co-reside with other machine language programs (the second includes keyboard-debounce and flashing cursor). Fitting requires soldering inside the computer and should only be carried out by an experienced hobbyist or technician. A fitting service is available in capital cities for only $20.00 and a list of installers is included with each kit. (Specify TRS-80 Model I or System 80 when ordering.)

All prices include Sales Tax and are correct at time of publication but are subject to changes without notice. All equipment carries MICRO-80's Australia-wide 90-day warranty covering parts and labour.

DISK OPERATING SYSTEMS & DEVELOPMENT SOFTWARE

You can increase your programming productivity, the execution speed and 'user friendliness' of your programs by using an enhanced Disk Operating System (DOS). Together with the other utility software, you can get the most from your disk drives.

DOSPLUS 3.3 **$99.95**
(Specify Model I single or double density or Model III)
An economic DOS intended for the first-time user and requiring single-sided disk drives. (The TRSDOS & DISK BASIC MANUAL is required to supplement the DOSPLUS manual).

DOSPLUS 3.4 **$149.95**
(Specify Model I single or double density or Model III)
With a high degree of compatibility with TRSDOS, DOSPLUS 3.4 supports single- or double-sided, single or double density, 5" or 8" disk drives with any track count (up to 96). Suitable for the first-time or experienced user wanting a fuss-free, bug-free, easy to understand but very powerful DOS which supports variable length records up to 255 bytes long. Comes with a stand alone manual.

ENHBAS **$52.95**
ENHBAS adds over 30 new commands and functions to your BASIC interpreter including high speed SORT, labels in BASIC, RESTORE to any line number, WHILE-WEND for structured programming, SCROLL, LEFT, INVERT, DRAW and PLOT to give you ease of control over graphics, SOUND and PLAY to add realistic sound effects and many more. Makes programming a breeze! Available for Model I or III, disk or cassette — specify which when ordering.

NEWDOS 80 VERSION 2.0 **$185.00**
(Specify Model I or Model III)
Newdos 80 suits the experienced user who has already used TRSDOS, understands the manual and is prepared to learn the somewhat complicated syntax of one of the most powerful DOS's available. With the correct hardware, Newdos 80 supports any mix of single- or double-sided, single or double density, 5" or 8" disk drives with track counts up to 96. It provides powerful, flexible file handling in BASIC including variable length records up to 4096 bytes. Definitely not for the beginner.

MASTER DISK DIRECTORY **$20.95**
FIND THE PROGRAM FAST!! PAYS FOR ITSELF BY RELEASING REDUDANT DISK SPACE!! MASTER DIRECTORY records the directories of all your individual disks onto one directory disk. Then it allows you examine them, find an individual file quickly, list files alphabetically, weed out redundant files, identify disks with free space, list files by extension, etc., etc. This program is invaluable for the serious disk user and will pay for itself many times over.

THE FLOPPY DOCTOR/MEMORY DIAGNOSTIC
Model I Disk **$36.50** Model III Disk **$43.50**
THE MICRO CLINIC offers two programs designed to thoroughly check out the two most trouble-prone sections of the TRS-80 — the disk system (controller and drives) and the memory arrays. Both programs are written in Z80 machine code and are supplied together on diskette for a minimum 32K, one disk system. Specify Model I or Model III.

MORE ENTERTAINMENT SOFTWARE

ADVENTURE HINT BOOK **$10.95**
If you can not go any further this will give you clues that may help — written by Scott Adams for Adventures 1—9.

LABYRINTH **$26.50**
Labyrinth — you move through a gigantic labyrinth and scattered through this nightmare are a multitude of objects and obstacles. A minotaur prowls the corridors — you must kill it before it kills you, Labyrinth has over 550 locations — be patient.

ASYLUM **$26.50**
Asylum places you in a cell, you have to escape. It's harder than it sounds, lots of hazards will be encountered.

DEATHMAZE 5000 **$26.50**
Deathmaze 5000 is another 3-D adventure. You move through a 5 storey building — your goal is to leave the deathmaze alive.

~~EPSON MX80 III?~~

NO! IT'S THE NEW

DT-80

with better performance than the MX-80!

PRICE ... not ~~$999~~ **BUT $699**

MICRO-80 HELPS YOU BEAT DEVALUATION!

The recent devaluation of the Australian dollar has increased the prices of imported printers. MICRO-80 helps you beat the price rises by offering a 5% discount on all prices.

NEW EPSON FX-80

160c.p.s., 6 character sizes proportional printing, up to 256 user-defined characters, dot matrix

PRICE: $1399

EPSON MX100 III

80 c.p.s., 132 col., bit graphics, dot matrix.

$1500 LIST PRICE $1580

ITOH PROWRITER 8510

120c.p.s., 80 col., bit graphics, proportional print dot matrix.

$1150 LIST PRICE $1244

ITOH PROWRITER 1550

120c.p.s., 132 col., bit graphics, proportional print dot matrix.

$1495 LIST PRICE $1573

MICROLINE 84

200c.p.s., 132 col., bit graphics, dot matrix.

$2110 LIST PRICE $2289

ITOH F10-40P

40c.p.s., 132 col., correspondence quality daisywheel.

$2535 LIST PRICE $2880

All prices include Sales Tax and are correct at time of publication but are subject to change without notice. All equipment carries MICRO-80's Australia-wide 90-day warranty covering parts and labour.

The text editing capability of the program was a major attraction when I was considering whether or not to spend the money, but it is somewhat of a disappointment. It is fine if you just need to change a word or two, or make minor alterations to the text, but a major change such as replacing a whole sentence or paragraph is quite a procedure. I have improved as I have become more used to the program but I would dearly have loved to see some editing features such as the Level 2 BASIC 'Delete', 'Hack' and 'Kill'.

The program also permits you to save documents to cassette and, naturally, retrieve them for later use. The big disadvantage of this procedure is that it is painfully slow and if you wanted to save a full document of 700 words, I suggest that you start the process and then go to the football or something whilst the computer grinds it all out onto the tape. For example, it just took 16 minutes 38 seconds to save the last three paragraphs of this review and I shudder to think how long a full document would take. Retrieval is quite a deal faster and also permits you to add the retrieved text to the document already in memory. If you had an application such as adding standard paragraphs to letters then I suppose that the time taken to get the standard paragraphs on to tape would be worthwhile.

WORP-1 also permits some pretty fancy tricks with the printer including varying the number of characters per line with automatic setting of the text about the centre of the page. It also does right justification (the alignment of the right hand margin of the text) but, again, right justification is painfully slow and I don't use it at all unless I have a very real need for the finished article to look pretty.

Generally, the program is an improvement over the old Remington but it is very slow and fairly limited in what it can do. To be fair, Dick does point out these shortcomings in the documentation but you don't really get an appreciation of just what "slow" and "limited" really mean until you have used the program for a while. I would think that the application would also have some bearing on its usefulness. If, for example, all that you needed a word processor for was short, sharp letters or memoranda, then I would think that WORP-1 would fill the bill. If, on the other hand, your requirements are more exotic, then I would think hard about continuing to save towards a machine language program such as SCRIPSIT or ELECTRIC PENCIL. If you have the correct application, and considering the cost of programs such as SCRIPSIT, WORP-1 does represent good value for money.

- 0000000000 -

***** MICROBUGS *****

Inevitably, no matter how careful one tries to be, there will be errors, mistakes and omissions in articles and programs. Here's where we make corrections.

DEF FN ADDENDUM

A number of readers have suggested ways to circumvent the problem that confronted one user regarding the DEF FN and FN statements.

The first method suggested was to omit the DEF FN statement altogether and replace all invocations of the function (FN expressions) by the expression that originally defined the function. This will work satisfactorily but can be tedious in the case of long and frequently used functions. For example, these program statements:

```
25 DEF FNA(X,Y,Z)=2*Y-Y*Y-5*Z
3Ø Y=2*A*FNA(I,J,K)
    could be replaced by:
(25 deleted)
3Ø Y=2*A*(2*I-J*J-5*K)
```

The second method is to replace the DEF FN statement by a subroutine and each function call by a subroutine call. However, care must be taken with this method to avoid any side-effects through corruption of variables used elsewhere in the program. The equivalent of the above program could be written as follows:

```
25 AØ=2*XØ-YØ*YØ-5*2Ø:RETURN
3Ø XØ=I:YØ=J:ZØ=K:GOSUB25:Y=2*A*AØ
```

This serves to demonstrate the power of the dummy variables that occur in function definitions - they have no effect whatsoever on the value of any real variables with the same name elsewhere in the program. In the case of the subroutine, it should be ensured that the variables used are unique and that their values are correctly initialized before the subroutine call. This

is actually similar to the way in which the function call FNA(I,J,K) is evaluated in practice.

JUMP THE RAPIDS ON THE MODEL III

This program uses the utility MOVIE originally published in the September, '81 issue. The MOVIE utility works by intercepting the DISK BASIC exits and positions itself below the start of BASIC program space. In the Model I and System 80, BASIC programs normally start at 42E9H and MOVIE loads to this address and sets the BASIC program area pointers to beyond it, thus protecting itself and alleviating the need to protect and use high memory. The program initialization changes the pointers, prints message and exits to BASIC via Ø6CCH. Unfortunately, this does not work in the case of the Model III for two reasons - the non-disk BASIC program area starts at 43E9H (with more low memory being used by the system) and the Ø6CCH ROM code has been changed and can no longer be used as an entry point to BASIC.

A little investigation has shown that the MOVIE source file can be relatively easily modified to make it work with the Model III, even with DISK BASIC loaded. Rather than setting the various pointers in the initialization code, the following method is suggested:

```
XOR  A
LD   HL,XXXX     (XXXX = Address for start of BASIC)
LD   (HL),A
INC  HL
LD   (4ØA4H),HL
CALL 1B5ØH       (enter NEW routine)
```

This should work for the Model I, System 80 and Model III in Level II BASIC, and in DISK BASIC after it has been initialized.

For re-entry to BASIC, the following should work with all the machines:

```
LD   BC,1A18H
JP   19AEH
```

This is the code executed at Ø6CCH in the Model I/System 80 ROM and is the preferred re-entry point. The alternative, as suggested by Tandy documentation, is JP 1A19H.

This month's cassette and disk editions contain amended versions of the MOVIE utility for the benefit of our Model III subscribers, who have not been able to use the JUMP THE RAPIDS program.

To load the cassette version, use the SYSTEM command and reply / (Enter) when the prompt returns. Disk users should do the following from DOS READY:

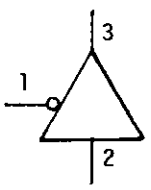
```
LOAD MOVIE/CMD      (Enter)
BASIC               (Enter)
SYSTEM              (Enter)
/28842              (Enter)
RUN"RAPIDS/BAS"     (Enter)
```

FREE SOFTWARE LIBRARY

The following changes to the program COMPOSER/BAS were omitted from the list of changes necessary to convert the cassette version to disk:

```
25 POKE&H8Ø36,Ø
DELETE 6ØØ - 64Ø
```

SYSTEM-80 CLOCK MODIFICATION - Vol. 3 Issue 9, August, 1982. There is an error in the circuit diagram shown in figure 1. Pin numbers 2 and 3 of IC3 (74LS367) are shown incorrectly and should be transposed. The corrected diagram is as follows:



Note that although the orientation of pins 1 and 2 of IC5 (74LS00) is different in figure 2 from that in figure 1, the operation of the circuit is functionally the same.

- 0000000000 -

**** SOFTWARE SECTION ****

***** LOAN CALCULATION PACKAGE - Peach and CC *****

This program, originally published in the June '82 issue, computes various figures relating to loans where interest is calculated on a daily reducing, capitalised monthly basis, e.g. building society housing loan.

The formula on which the program is based is R=L/A where:

R = repayments per month
L = loan amount
A = $(1-V^T)/I$
V = 1/(1+I)
T = term of loan in months
I = interest rate/1200

There are five separate calculations available. They are:

1. Repayment Calculation Given the amount borrowed, interest rate and term of loan in years, it will calculate the monthly repayment and the approximate total interest which would be paid over the full term of the loan. (This section incorporates a facility to include insurance instalments with loan repayments. If not required, the amounts can be entered as zero or the relevant program lines deleted).

2. Remaining term Calculation Given the current loan balance, interest rate and amount of monthly repayment, this will calculate the length of time remaining until the loan would be paid out (e.g. If you decide to pay an extra $30.00 per month, how long would the loan then run?).

3. Remaining Balance Calculation Given the current loan balance, interest rate and amount of monthly repayments, this section will calculate the balance remaining after a given period of time.

4. Dissection of repayments Given the amount of the loan, interest rate, monthly repayment and term of loan, this section shows how much of each repayment is interest and how much goes toward reducing the principal. It also calculates total interest to date year by year and shows loan balance (principal) month by month.

5. Repayment Factor Calculation This section calculates a repayment 'factor' for a given interest rate. The factor is the amount of repayment per month per $1,000.00 borrowed, e.g. for interest rate of 12.75% and term of 25 years the factor is 11.09 so the monthly repayment for a loan of say $32,000 over 25 years is 32 * 11.09 or $354.88 (this section was included as a source of factors for a ready-reckoner used at my place of work).

Being a daily reducing interest calculation, the interest charged will depend on the number of days between repayments, and to provide an acceptable average, 30.4167 is used as the number of days per month. This is 365/12.

Sections 1, 2, 3 and 4 assign two variables to each figure entered by the user. Calculations are then done using only one of each pair of variables so that having entered your particular figures once in any of the above sections, it is only necessary to hit ENTER or RETURN in response to the input statements in any other section if you wish to do further calculations using the same input data.

The program when RUN asks the user to enter figures for Loan amount, Interest rate, etc. and these should be entered without dollar signs, commas, percent symbols etc.

e.g. $32,000.00 - enter as 32000
12.75% - enter as 12.75

Remaining term and remaining balance calculations take a few seconds to arrive at the answer, especially for longer terms - just be patient.

- 0000000000 -

***** BIORYTHMS - Peach and CC *****

For some time, scientists have acknowledged the existence of biological rhythms in living organisms. Some people believe that three such biological cycles commence from the date of birth in humans and continue throughout life. In addition, they also suggest that these cycles (known as the

Physical, Emotional and Intellectual) are related to the way we feel and to our behaviour. This program, originally published in the February '80 issue, plots the three curves relating to the theory of Biorhythms.

The user is prompted for his or her date of birth and the current date. After a moment taken to calculate the relative phases of the cycles, the curves are drawn on the high resolution graphics screen. On the Tandy colour computer, the description of the curves is on the text screen while the curves themselves are on the graphics screen, but you may toggle from one to the other by pressing any key (except 'A' which restarts and prompts for new input).

The Hitachi can mix text and graphics on its high resolution screen. To terminate either program, you press the 'BREAK' key.

- 0000000000 -

***** MOVE BY 1's - LII/16K by Steven and Andrew Ludman *****

This program was originally written on a Level I machine and later rewritten on a Level II machine. This means that Level I users can also type in the program providing the obvious changes are made, i.e. PRINT@ changed to PRINT AT or P.A., etc.

The program draws one of five tracks and the objective is to move from the starting position to the end (it is up to you not to cheat!) in as few moves as possible.

To move around the track, you input a horizontal and a vertical move. This is done in lines 100 to 120. Line 130 checks to see if you are quitting.

The rules of moving are that you must not leave the borders, nor are you allowed to land on anything. But more importantly, your horizontal OR your vertical (not both) must differ from your previous move by one (hence the title MOVE BY 1's!)

Checking this is done in lines 160 and 170. Variables P,R are your previous moves while M,V are the inputted moves. For a legal move, either M+V must equal P+V-1 or P+R+1.

Lines 200 and 210 check to see that your new moves stay within the border.
Line 240 checks to see if there is anything at the points where you land.
Lines 270 to 290 set up a loop checking for set points in the landing (winning) area. If there is any point set, then the program breaks out of the loop and goes to line 300.

The subroutines to draw the tracks are from line 570 onwards.

A subroutine to draw the boundary is also used (lines 450-540).

N.B. Quite a few interesting loops are used for drawing obstacles showing how versatile loops and sets can be!

- 0000000000 -

***** CHECK SUM - LII m.l. by B.N. Briggs *****

This short program loads into the buffer on a Level 2 machine, Auto starts, PEEKs at the Memory Size and then relocates to reside below that point, resetting the memory size to protect itself.

The program performs two functions:

(a) It calculates and displays the length of any BASIC program in memory.

(b) It performs a checksum of that program, starting at the beginning of the second line to the end of the BASIC program and adding together the value of each byte.

Note: Starting at the second line enables you to use the first line to contain the length and checksum, and provided the length of the first line is not changed, alter these without altering the checksum. (Altering the length will alter the checksum, however).

ENTERING THE PROGRAM

The program may be entered using an Editor/Assember, or a Utility such as RSM 2. Note the source shows two Origins. This is required to leave a gap in the middle for the stack. The stack uses the buffer when loading System Tapes and then punches a tape using

Start	End	Entry	Name
41E2H	42DAH	41E2H	CHKSUM

METHOD OF OPERATION

ENTER and LOAD as you would any System Tape. After the program is loaded, it will Auto start and relocate to the top of Memory. The routine patches into 41B2H, which is one of the DOS exit points for BASIC. When you enter anything from the keyboard, it is first compared with the computer's own Reserved Word Table and then jumps to 41B2H before returning to ROM. By entering a jump at 41B2H, we cause it to jump to "Checksum" and compare the word entered with the word located at "Table" (See Source listing). If it finds a match it will carry on with the routine, or, if not found, it returns to ROM.

TO USE "CHECKSUM"

Enter in as the first line of your BASIC Program
REM Program length = xxxxx Checksum = xxxxx
Now enter "Check" and the program length and checksum will be displayed.
Edit the first line and change the "xxxxx" to the displayed figures and CSAVE your program. Now, whenever you reload that program, load in "Checksum" as well. (By having "Checksum" load into the buffer, means that it can be loaded at any time, without upsetting your BASIC Program, unless you have insufficient room at the Top of Memory).

If the program length and Checksum match the first line of your program, you may be sure that you have had a successful load. Beats doing a CLOAD? on a 10K or more program !!!

- 0000000000 -

***** MATRIX MANIPULATION - LI/4K (C) Martin Downey *****

FEATURES:
* PRINT, MULT, ADD/SUB, INVERSE commands.
* Handles up to 10x10 and larger.
* Stand-alone or use in your own programs.

TO LOAD: Matrix Manipulation is a BASIC program so it is loaded using the CLOAD command. If the program fails to load successfully try again, perhaps with a different volume setting.

After the program has loaded type in RUN and press ENTER. The title will be displayed along with the SIX COMMANDS (MATRIX INSTRUCTIONS). Initially there will be 0 MATRICES ALREADY DEFINED. To use a particular INSTRUCTION simply type in the corresponding number (1-6) and press ENTER. The SIX commands are described below.

PRINT: This command will print out all the matrices that have been DEFINEd or created using MULT etc. The matrices will be printed one at a time. Pressing ENTER will display the next matrix. Very large matrices (more than 10 columns) may wrap around spoiling the display.

DEFINE: Allows you to input the matrices to be manipulated. You will first be asked to input the size of the matrix. Type in the number of rows then a comma then the number of columns then press ENTER. If you only input one number an error message "WHAT?" will be displayed telling you to input TWO numbers. This sort of error should be avoided as it will mess up the display. After entering the size you type in each element of the matrix followed by ENTER. The matrix will be displayed in its normal format as the elements are entered. When finished control will return to the SIX COMMANDS.

MULT: Allows multiplication of TWO MATRICES or a MATRIX and a CONSTANT. Normal laws of Matrix multiplication apply and if the two matrices are not of correct dimensions then a 'NOT POSSIBLE' will be displayed. Enter the number of the first matrix. Then enter a 1 or a 2 depending on whether you want to multiply by a constant or another matrix. Finally enter the CONSTANT or MATRIX NUMBER accordingly. The resulting matrix will then be displayed (and is automatically saved with the other matrices).

ADD/SUB: After entering the number of the first matrix you will be asked to enter a 'MULTIPLE OF ADDITION'. Entering a 1 will give straight addition, entering a -1 will give straight subtraction, but you can enter any other number of the SECOND MATRIX and the result will be displayed (and saved). Again, normal matrix rules apply.

INVERSE: Enter the matrix number (matrix must be square, of course) and after a short delay the inverse will be printed (and saved). If there is no inverse then 'SINGULAR' will be displayed. In either case THE ORIGINAL WILL BE DELETED so you might want to save another copy (use MULT times a constant 1).

DELETE: This command allows you to get rid of matrices that you may not want any more. This may become necessary if you get a message 'INSUFFICIENT MEMORY. TRY DELETE.'. (Note:

All matrices are stored in the single dimension array A). You will have to keep this in mind if you wish to use parts of the program in your own program (see below). When you DELETE a matrix all higher-numbered matrices will be moved down one. So if you DELETE matrix number 3, say, then matrix 4 will become 3, matrix 5 will become 4 and so on. You can check on the new matrix numbers using the PRINT command.

All the MATRIX COMMANDS are divided into subroutines so you can use them in your own programs. This will, however, require some modification so the following may be of help.

200-250	PRINT command
400-450	DEFINE command.
800-950	MULT command. Also uses part of PRINT and SUB 4000.
1000-1090	ADD/SUB command. Also uses part of PRINT and SUB 4000.
2000-3020	INVERSE command. Also uses part of PRINT and DELETE and SUB 4000.
4000-4020	SUB used by MULT, ADD/SUB, INVERSE and DELETE.
5000-5020	DELETE command. Also uses SUB 4000.
6000	Displays 'INSUFFICIENT MEMORY' flag. Used by MULT and ADD/SUB.

All matrices are stored in the single dimensioned array A. Matrices are stored sequentially from MATRIX 1 up. The first number is the number of ROWS then comes the number of COLUMNS followed by the matrix ELEMENTS. (NOTE: The number of elements equals ROWS * COLUMNS).

If you have any problems, suggestions or comments, please contact the author.

- 0000000000 -

***** AN ALTERNATIVE TO THE INPUT COMMAND - LII/4K-16K (C) K. Shillito *****

The '80 INPUT command suffers from a number of deficiencies, to wit:-

1. The comma and colon cannot readily be input.
2. The prompt always contains a ? which is not always wanted.
3. It requires the ENTER/NEWLINE key, even if the maximum input string length is known.

An alternative to the input command then should have the following properties:-

1. It should give extremely fast response.
2. It should not "BOUNCE" - this rules out routines which PEEK at keyboard memory.
3. It must allow prompts of any form.
4. It much echo to the screen, like input but unlike INKEY$.
5. It must automatically return when a specified number of characters is reached or when ENTER/NEWLINE is pressed.
6. It must allow backspace and erase in the usual fashion.

Well, the subroutine from lines 160 to 290 fills the bill; it also completely debounces the keyboard while allowing lightning fast response. NOTE THE FOLLOWING:-

(a) The variables I-M should be integers. Note the use of LET and variables for extra efficiency.
(b) The application in this program is only a beginning to the possible uses of this subroutine.

HOW IT WORKS

Line 170 Set up a null string A$ which will hold the input.
Line 180 Clear the VDU area which will hold the echo.
Lines 190-200 The "heart" of the subroutine, particularly part (a) thereof below:

(a) A = A + INKEY$: We cycle around and around, adding a character whenever a key is pressed. Fortunately for this program INKEY$ will not return another value until the key is released. Note that even control keys will return a value to INKEY$, viz:

	ENTER	returns CHR$(13)
SHIFT	←	returns CHR$(24)
	→	returns CHR$(8)

(values below 32 other than the above are assumed to be → , since pressing other arrows or CLEAR is silly).

(b) PRINT@J,A : This echoes A$ to the screen.
(c) IF LEN(A)<I...: This stops line 210 from accepting further input once the specified length is reached. The subroutine cannot yet RETURN, however, since the last character might be a control character.
(d) ...AND RIGHT$(A,L)> B : This temporarily stops further input if a control character is pressed, so that the control character might be chopped off A$ and acted upon.

KEY DEBOUNCE

Since INKEY$ freezes until the key is released, and then lines 210 to 220 are executed, this turns out to be just the right speed to debounce the keyboard (incidentally, this is the only non-machine language debounce method I know of).

Line 220	Is needed because of the way that RIGHT$(A,L) acts on null strings.
Line 230	The final character of A$'s ASCII value is called M. We now act depending on the value of M.
Line 240	If M>31, it is not a control character, so as mentioned under line 210(c) the subroutine can now RETURN.
Line 250	If M = 24, the SHIFT ← was pressed, so we start again.
Line 260	We chop off the control character from A$.
Line 270	If M = 13, then RETURN was pressed (before A$ reached the maximum permitted length), so whatever is in A$ is RETURNed.
Lines 280-290	We assume ← was pressed, so if LEN(A)>Ø, we chop off a character, and continue to accept input.

- 0000000000 -

***** FLEXITIME - LII/4K by David Grigg *****

This is for all you public servant types, or anyone else who works on the system of flexible working hours. Adding up or subtracting all of those hours and minutes at the end of the week (or, in my case, the month!) to figure out if you owe your boss some hours or if he owes you some time off can be a definite pain in the neck. Quick, 12 hours 14 minutes less 3 hours 27 minutes!?! The program would be especially valuable to anyone working in a personnel department who has to check other people's attendance forms.

The working of the program is really quite straightforward, and explained in the inbuilt instructions. A particularly valuable feature is the ability of the program to handle mixed 12-hour and 24-hour formats.

My standard working day is 7 hours and 21 minutes long. You can tell that I work for the public service, can't you? Should you work a day which is a more reasonable length, say 7 hours or 7 hours 30 minutes, you must change the variable ST in line 20. The program changes all times on entry into minutes only. So, for a 7 hr. 30 minute day, change line 20 to read "ST = 7*60 + 30".

I hope you discover, as I did, during the past three months that you hadn't added up your flexitime forms, you've gained 10 hours extra credit!

- 0000000000 -

***** CODEBREAKER - LII/4K by David Grigg *****

This program was written specifically to enable me to solve those cryptograms (secret messages in a simple substitution code) which appear each month in GAMES magazine. But children, and many adults, find solving such code messages great fun, and a real intellectual challenge, whatever their source. Remember such things as Captain Marvel's code rings?

The CODEBREAKER program takes almost all the drudgery out of solving simple codes, and leaves you with all of the intellectual fun. It will speed up enormously the solution of messages in any simple cipher - one in which, say, the letter X always represents, say, the real letter A.

To use the program, you type in the coded message you want to solve (up to a maximum of six screen lines), including any punctuation marks in the message EXCEPT commas and colons. To end the message, enter the symbol # at the start of a new screen line. The program will then ask you if you wish it to carry out an analysis to see which are the most common code letters in the message. If you wish, it will then substitute for these code letters the most likely plaintext equivalents. For example, the most common letter will be assumed to be E, and so on. If this inspired computer guess is wrong, it can easily be changed.

You may then work on the cryptogram on the screen, making trials of letter substitutions, with the computer speeding up the trials. To try a guess that code letter X equals plaintext letter A, for example, you merely enter "X/A", and the computer carries out all the substitutions. If you had previously guessed that code Y equalled A, the program will automatically reset this guess.

- 000000000 -

```
10 '  ***LOANCALC***
20 'A LOAN CALCULATION PACKAGE
 FOR THE TRS80C. BASED ON A
 PROGRAM BY K. GLASSON.
30 CLS:PRINT@36,"L O A N   P A C
 K A G E"
40 PRINT@68,"==================
====="
50 PRINT@104,"** M E N U **"
60 PRINT:PRINT"  1: REPAYMENT
70 PRINT"  2: REMAINING TERM
80 PRINT"  3: REMAINING BALANCE
90 PRINT"  4: DISSECTION OF REPA
YMENTS
100 PRINT"  5: REPAYMENT FACTOR
110 PRINT"  6: END PROGRAM
120 PRINT:PRINT"  WHICH FUNCTION
 ? ";
130 A$= INKEY$:IFA$=""THEN130
140 G= VAL(A$):IFG>0ANDG<7 THEN
PRINTA$ ELSE 130
150 ON G GOTO160,340,410,490,660
,760
160 CLS:PRINT"  LOAN REPAYMENT C
ALCULATION"
170 PRINT"  ==================
======"
180 GOSUB770
190 INPUT" TERM OF LOAN IN YEARS
 ";T:T=T*12
200 INPUT" INSURANCE PREMIUM-LIF
E ";M
210 INPUT" INSURANCE PREMIUM-FIR
E ";F:P=M+F
220 R=L*I/(1-(1+I)^-T)
230 F1$="MONTHLY PAYMENT:$$###.#
#"
240 F2$="+$$##.##OR$$###.##"
250 F3$=" ($$###.## PER WEEK)
260 IFP=0THEN290
270 PRINT:PRINTUSING F1$+F2$+F3$
;R,P/12,R+P/12,(R*12+P)/52
280 GOTO300
290 PRINT:PRINTUSING F1$;R:PRINT
USING F3$;R*12/52
300 PRINT:PRINTUSING"TOTAL INTER
EST PAID:$$###,###.##";R*T-L
310 PRINT@481,"          mENU  O
R  eND ?";
320 B$= INKEY$:IFB$=""THEN320
330 IF B$="E"THEN760 ELSE 30
340 CLS:PRINT"  REMAINING TERM C
ALCULATION"
350 PRINT"  ==================
======"
360 PRINT:PRINT:INPUT" BALANCE
OF LOAN ";B
370 GOSUB780
380 INPUT" MONTHLY REPAYMENT ";R
390 T=INT(-LOG(1-B*I/R)/LOG(1+I)
)
400 PRINT@256,"REMAINING TERM IS
 ";:PRINT USING"##.##";T/12;:PRI
NT" YEARS    ("T" MONTHS )":PRIN
T:GOTO310
410 CLS:PRINT" REMAINING BALANCE
 CALCULATION"
420 PRINT" ==================
========":PRINT:PRINT
430 GOSUB770
440 INPUT" MONTHLY REPAYMENT ";R
450 INPUT" PERIOD IN YEARS ";T:M
=12*T
460 K=(1+I)^-M:L=(L-R/I)/K+R/I
470 PRINT@288,"BALANCE REMAINING
 AFTER"T"YRS":PRINTUSING"IS $$###
#,###.##";L
480 GOTO310
490 CLS:PRINT"DISSECTION OF LOAN
 REPAYMENTS"
500 PRINT"==================
=======":PRINT:PRINT
510 GOSUB770
520 INPUT" MONTHLY REPAYMENT ";R
530 C1=0
540 CLS:PRINT" NO INTEREST PRINC
IPAL  BALANCE
550 FORT=1TO9999
560 C=L*I:L=L+C-R:C=(INT(C*100))
/100:C1=C1+C
570 PRINTUSING"### #,###.## #,##
#.## ###,###.##";T,C,R-C,L;
580 IFL<=R THEN PRINTUSING" AND
NOW ONLY $$###.## MORE.    TOTAL
 INTEREST PAID:$$####,###.##";L,C
1+L*I;:GOTO310
590 IFT/12<>INT(T/12) THEN640
600 IFT=1THEN S$=""ELSE S$="S"
610 PRINTUSING"##.#% OF LAST PAY
MENT OFF BAL.";(R-C)/R*100:PRINT
USING"INTEREST TO DATE:$$####,###
.##";C1
620 GOSUB800
630 CLS:PRINT" NO INTEREST PRINC
IPAL  BALANCE
640 NEXT
650 CLS:GOTO310
660 CLS:PRINT"REPAYMENT FACTOR C
ALCULATIONS"
670 PRINT"==================
======="
680 PRINT:GOSUB780
690 L=1000:PRINT@64,"TERM
  FACTOR (@"I*1200"%)":PRINT
700 FORF=1TO999
710 R=L*I/(1-(1+I)^-(12*F))
720 PRINTF,:PRINT USING"##.##";R
730 IF F/10=INT(F/10) THEN GOSUB
800:PRINT@96,""
740 NEXTF
750 GOTO310
760 CLS:PRINT"THANK YOU":END
770 INPUT" LOAN AMOUNT ";L
780 INPUT" INTEREST RATE ";I:I=I
/1200
790 RETURN
800 PRINT@481,"cONTINUE, mENU OR
 eND ?";
810 B$=INKEY$:IFB$=""THEN810
820 IFB$="C"THEN RETURN
830 IFB$="E"THEN760ELSE30
```

**** BIORHYTHMS ****

COLOUR COMPUTER

```
10 ' ***BIORYTHMS***
20 'ADAPTED FOR THE TRS80C
30 'FROM A TRS80 PROGRAM BY
B. SIMPSON
40 PMODE4,1
50 CLS:PRINT@2,"THE  THEORY  OF
 BIO-RHYTHMS"
60 PRINT:PRINT"RESEARCHERS BELIE
VE THAT FROM   BIRTH, 3 DIFFEREN
T CYCLES START TO OPERATE IN YOU
R BODY, CALLED BIO-RHYTHMS,WHICH
 INFLUENCE YOUR";
70 PRINT"THOUGHTS, FEELINGS AND
ACTIONS. KNOWING THE STATE OF TH
E 3CYCLES(PHYSICAL,EMOTIONAL,INT
ELLECTUAL) AT GIVEN TIMES CAN";
80 PRINT" PREPARE YOUFOR YOUR CH
ANGING MOODS."
90 PRINT:PRINT"THE TRS80C  WILL
PLOT YOUR 3 BIO-RHYTHM CURVES FO
R YOU TO STUDY.":PRINT:INPUT"PRE
SS 'ENTER' TO CONTINUE";X
100 CLS:SCREEN0:PRINT
110 PRINT" TODAY'S DATE
120 INPUT" (DD,MM,YY) ";TD,TM,TY
130 IFTM<1ORTM>12GOTO180
140 FORL=1TO TM:READ MT:NEXT:RES
TORE
```

```
150 DATA 31,28,31,30,31,30,31,31
,30,31,30,31,31
160 IFTM=2THENIF TY/4=INT(TY/4)
THEN MT=29
170 IFTD>0ANDTD<=MT THEN190
180 PRINT"  HA HA, VERY FUNNY. T
RY AGAIN":GOTO110
190 PRINT:PRINT" YOUR BIRTH DATE
200 INPUT" (DD,MM,YY) ";BD,BM,BY
210 IFBM<1ORBM>12THEN270
220 D1=TD+(TM*100)+(TY*10000):D2
=BD+(BM*100)+(BY*10000)
230 IFD2>D1 THEN270
240 FORL=1TOBM:READMT:NEXT:RESTO
RE
250 IFBM=2THEN IF BY/4=INT(BY/4)
 THEN MT=29
260 IFBD>0ANDBD<=MT THEN280
270 PRINT"  HA HA, VERY PECULIAR
. TRY AGAIN":GOTO190
280 CLS:PRINT"THE PROGRAM WILL D
RAW THE THREE CURVES AS FOLLOWS:
-"
290 PRINT:PRINT"PHYSICAL -SOLID
LINE"
300 PRINT"EMOTIONAL  -DOTTED LIN
E"
310 PRINT"INTELLECTUAL -DASHED L
INE"
320 PRINT:PRINT"A GRAPH OF 28 DA
YS (4 WEEKS)"
330 PRINT"IS SHOWN -BEGINNING WI
TH TODAY."
340 PRINT:PRINT"ABOVE THE CENTRE
LINE IS"
350 PRINT"ACTIVE, BELOW IS 'PASS
IVE', AND"
360 PRINT"ON IT IS 'CRITICAL'.
370 PRINT:PRINT"HIT 'A' TO ENTER
 NEW DATES, OR"
380 PRINT"ANY OTHER KEY TO SWAP
SCREENS.";
390 D=TD:M=TM:Y=TY:GOSUB650:D1=D
S
400 D=BD:M=BM:Y=BY:GOSUB650:D2=D
S
410 D3=D1-D2+1
420 P=D3-(INT(D3/23)*23)
430 E=D3-(INT(D3/28)*28)
440 I=D3-(INT(D3/33)*33)
450 PCLS
460 I$=INKEY$:IFI$=""THEN460
470 SCREEN1,1
480 FORL=0TO4:LINE(63*L,0)-(63*L
,180),PSET:NEXT
490 LINE(0,180)-(253,180),PSET:L
INE(0,90)-(253,90),PSET
500 FORL=0TO252STEP9:PSET(L,89):
PSET(L,91):PSET(L,179):PSET(L,17
8):NEXT
510 FORL=1TOTM:READMT:NEXT
520 IFTM=2ANDTY/4=INT(TY/4) THEN
 MT=29
530 FORL=487TO508STEP7:TD=TD+7
540 IFTD>MT THENTD=TD-MT:TM=TM+1
550 IFTM>12THEN TM=1
560 C=1:F=23:N=P:GOSUB710
570 C=2:F=28:N=E:GOSUB710
580 C=3:F=33:N=I:GOSUB710
590 I$=INKEY$
600 TE=-1
610 I$=INKEY$:IFI$=""THEN610
620 IFI$="A"THEN100
630 TE=NOT TE:IF TE THEN SCREEN0
 ELSE SCREEN1,1
640 GOTO610
650 'CALC "DAYS FROM 1900" ROUTI
NE
660 Y1=Y:IFM<3 THEN Y1=Y1-1
670 DS=Y*365+INT(Y1/4):IFM=1THEN
690
680 FORL=1TOM-1:READMT:DS=DS+MT:
NEXT:RESTORE
690 DS=DS+D:RETURN
700 'SINE CURVE ROUTINE
710 IN=.1115:DC=360*.0174533:FOR
L=0TO252
720 IF C=3 AND L/4=INT(L/4) OR C
=2 AND L/2=INT(L/2) OR C=1THEN Y
C=SIN(N/F*DC)*90
730 PSET(L,90-YC)
740 N=N+IN:NEXT:RETURN
```

```
        **** LOAN CALCULATION ****

              HITACHI PEACH


10 ' LOAN CALCULATION PACKAGE FOR THE HI
TACHI PEACH
20 ' BASED ON A PROGRAM BY K. GLASSON
30 CLS:LOCATE26,5:PRINT"L O A N   P A C
K A G E"
40 LOCATE26,6:PRINT"==================
==="
50 LOCATE31,8:PRINT"** M E N U **"
60 PRINT:PRINTTAB(25)"1.  REPAYMENT CALC
ULATION"
70 PRINTTAB(25)"2.  REMAINING TERM CALCU
LATION"
80 PRINTTAB(25)"3.  REMAINING BALANCE CA
LCULATION"
90 PRINTTAB(25)"4.  DISSECTION OF REPAYM
ENTS"
100 PRINTTAB(25)"5.  REPAYMENT FACTOR CA
LCULATION"
110 PRINTTAB(25)"6.  END PROGRAM"
120 PRINT:PRINTTAB(29)"WHICH FUNCTION ?"
;
130 A$=INKEY$:IFA$=""THEN130
140 G=VAL(A$):IFG>0ANDG<7THENPRINTA$ELSE
130
150 ON G GOTO160,340,410,490,660,760
160 CLS:PRINT"  Loan Repayment Calculati
on"
170 PRINT"  ========================="
180 GOSUB770
190 INPUT" Term of Loan in Years ";T:T=T
*12
200 INPUT" Insurance Premium < Life > ";
M
210 INPUT" Insurance Premium < Fire > ";
F:P=M+F
220 R=L*I/(1-(1+I)^-T)
230 F1$="Monthly Payment : $###.##"
240 F2$=" + $##.## or $###.##"
250 F3$=" ($###.## per week)"
260 IFP=0THEN290
270 PRINT:PRINTUSING F1$+F2$+F3$;R,P/12,
R+P/12,(R*12+P)/52
280 GOTO300
290 PRINT:PRINTUSING F1$;R:PRINTUSING F3
$;R*12/52
300 PRINT:PRINTUSING"Total Interest Paid
 : $###,###.##";R*T-L
310 LOCATE25,24:PRINT"Menu or End <M/E>
?";
320 B$=INKEY$:IFB$=""THEN320
330 IFB$="E"THEN760ELSE30
340 CLS:PRINT  "Remaining Term Calculati
on"
350 PRINT"  ========================="
360 PRINT:PRINT:INPUT" Balance of Loan "
;B
370 GOSUB780
380 INPUT" Monthly Repayment ";R
390 T=INT(-LOG(1-B*I/R)/LOG(1+I))
400 PRINTUSING"Remaining Term is ##.## y
ears (### months)";T/12,T:PRINT:GOTO310
```

```
410 CLS:PRINT" Remaining Balance Calcula
tion"
420 PRINT" ============================
":PRINT:PRINT
430 GOSUB770
440 INPUT" Monthly Repayment ";R
450 INPUT" Period in Years ";T:M=12*T
460 K=(1+I)^-M:L=(L-R/I)/K+R/I
470 PRINTUSING"Balance remaining after #
# years is $####,###.##";T,L
480 GOTO310
490 CLS:PRINT"Dissection of Loan Repayme
nts"
500 PRINT"============================"
:PRINT:PRINT
510 GOSUB770
520 INPUT" Monthly Repayment ";R
530 C1=0
540 CLS:PRINT" No.     Interest       Prin
cipal        Balance"
550 FORT=1TO9999
560 C=L*I:L=L+C-R:C=(INT(C*100))/100:C1=
C1+C
570 PRINTUSING" ###     #,###.##        #,
###.##      ###,###.##";T,C,R-C,L
580 IFL<=R THENPRINT:PRINTUSING" and onl
y $####.## more.   Total Interest Paid :
$####,###,##";L,C1+L*I:GOTO310
590 IFT/20<>INT(T/20)THEN640
600 IFT=1THENS$=""ELSES$="s"
610 PRINT:PRINTUSING"##.#% of last payme
nt off Balance.   Interest to date : $##
#,###.##";(R-C)/R*100,C1
620 GOSUB800
630 CLS:PRINT" No.     Interest       Prin
cipal        Balance"
640 NEXT
650 CLS:GOTO310
660 CLS:PRINT"Repayment Factor Calculati
ons"
670 PRINT"============================"
680 PRINT:GOSUB780
690 L=1000:PRINT"Term           Factor (@
"I*1200"%)":PRINT
700 FORF=1TO999
710 R=L*I/(1-(1+I)^-(12*F))
720 PRINTUSING" ###             ###.##";F,
R
730 IF F/15=INT(F/15)THENGOSUB800:LOCATE
0,6:PRINTCHR$(26);
740 NEXTF
750 GOTO310
760 CLS:PRINT"Thank You":END
770 INPUT" Loan Amount ";L
780 INPUT" Interest Rate ";I:I=I/1200
790 RETURN
800 LOCATE20,24:PRINT"Continue, Menu or
End <C/M/E> ?";
810 B$=INKEY$:IFB$=""THEN810
820 IFB$="C"THENRETURN
830 IFB$="E"THEN760ELSE30
```

****** BIORHYTHMS ******

HITACHI PEACH

```
5 ' BIORHYTHM CURVE PLOTTER FOR THE HITA
CHI PEACH
7 ' ORIGINAL AUTHOR: BERNIE SIMSON
10 CLS:LOCATE23,4:PRINT"THE  THEORY  OF
 BIO-RHYTHMS":LOCATE23,5:PRINT"---------
--------------------"
20 PRINT:PRINT"Researchers believe that
from birth, 3 different cycles start to"
:PRINT"operate in your body, called BIO-
RHYTHMS, which influence your"
30 PRINT"thoughts, feelings and actions.
  Knowing the state of the 3":PRINT"cycl
es (Physical, Emotional, Intellectual) a
t given times can"
40 PRINT"prepare you for your changing m
oods."
50 PRINT:PRINT" The PEACH will plot your
 3 BIO-RHYTHM curves for you to study.":
PRINT:INPUT"Press 'RETURN' to continue";
X$
80 CLS:INPUT"Name of person to be analys
ed ";P$
100 INPUT"Today's Date (DD,MM,YY) ";TD,T
M,TY
110 IFTM<1ORTM>12GOTO160
120 FORL=1TOTM:READMT:NEXT:RESTORE
130 DATA 31,28,31,30,31,30,31,31,30,31,3
0,31,31
135 IFTM=2THENIFTY/4=INT(TY/4)THENMT=29
140 IFTD>0ANDTD<=MTGOTO170
160 PRINT"  HA HA, VERY FUNNY. TRY AGAIN
":GOTO100
170 INPUT"Your Birth Date (DD,MM,YY) ";B
D,BM,BY
180 IFBM<1ORBM>12GOTO240
182 D1=TD+(TM*100)+(TY*10000):D2=BD+(BM*
100)+(BY*10000)
184 IFD2>D1GOTO240
190 FORL=1TOBM:READMT:NEXT:RESTORE
195 IFBM=2THENIFBY/4=INT(BY/4)THENMT=29
200 IFBD>0ANDBD<=MTGOTO250
240 PRINT"  HA HA, VERY PECULIAR. TRY AG
AIN":GOTO170
250 D=TD:M=TM:Y=TY:GOSUB2000:D1=DS
260 D=BD:M=BM:Y=BY:GOSUB2000:D2=DS
270 D3=D1-D2+1
280 P=D3-(INT(D3/23)*23)
290 E=D3-(INT(D3/28)*28)
300 I=D3-(INT(D3/33)*33)
390 CLS
400 LINE(4,24)-(4,144),PSET
410 FORL=4TO480STEP20:LINE(L,84)-(L+10,8
4),PSET:NEXTL
414 L=(51-LEN(P$))\2
415 LOCATE L,0:PRINT"* * *  ";P$;"'S  BI
O-RHYTHMS  * * *";
420 FORL=124TO484STEP120:FORZ=24TO138STE
P6:LINE(L,Z)-(L,Z+3),PSET:NEXTZ,L
480 FORL=0TO4:LOCATE15*L,18:PRINT"^";:NE
XT
490 LOCATE0,19:PRINT"Today";
500 FORL=1TOTM:READMT:NEXT
520 FORL=1TO4:TD=TD+7
540 IFTD>MTTHENTD=TD-MT:TM=TM+1
560 IFTM>12THENTM=1
580 LOCATE15*L-1,19:PRINTTD;"/";TM;
600 NEXT
620 LOCATE8,22:PRINT"Physical";:FORL=140
TO180STEP2
640 PSET(L,180):NEXT
660 LOCATE29,22:PRINT"Emotional";:FORL=3
20TO360STEP3
680 PSET(L,180):NEXT
700 LOCATE51,22:PRINT"Intellectual";:FOR
L=520TO560STEP6
720 PSET(L,180):NEXT
740 C=1:F=23:N=P:GOSUB2500
760 C=2:F=28:N=E:GOSUB2500
780 C=3:F=33:N=I:GOSUB2500
800 LOCATE65,5:PRINT"Active";:LOCATE65,6
:PRINT"Stage";
820 LOCATE65,14:PRINT"Passive";:LOCATE63
,15:PRINT"Stage";
840 LOCATE63,10:PRINT" Critical";
860 GOTO860
2000 ' CALC "DAYS FROM 1900" ROUTINE
2020 Y1=Y:IFM<3THENY1=Y1-1
2030 DS=Y*365+INT(Y1/4):IFM=1GOTO2050
2040 FORL=1TOM-1:READMT:DS=DS+MT:NEXT:RE
STORE
2050 DS=DS+D:RETURN
2450 ' SINE CURVE ROUTINE
2500 IN=.06:DC=360*.0174533:FORL=4TO484
```

```
2520 IFC=3THENIFL/6=INT(L/6)GOTO2560ELSE
2800
2540 IFC=2THENIFL/3=INT(L/3)GOTO2560ELSE
2800
2560 YC=SIN(N/F*DC)*60
2580 PSET(L,84-YC)
2800 N=N+IN:NEXT:RETURN
```

****** LII/16K MOVE BY 1'S ******

TRS-80/SYSTEM-80

```
10 REM   LENGTH =  6615 : CHECKSUM = 60795
20 CLS:PRINTTAB(20)"M O V E  B Y  1 ' S":PRINT@960,"";
30 INPUT"DO YOU WANT INSTRUCTIONS (Y/N)";A$
40 N=0:IFA$="Y"THENGOSUB320
50 CLS:PRINT"THERE ARE 5 POSSIBLE TRACKS TO CHOOSE FROM":PRINT
60 PRINT"1 - SIMPLE":PRINT"2 - EASY":PRINT"3 - NOT SO EASY":PRIN
T"4 - DIFFICULT":PRINT"5 - QUITE HARD"
70 PRINT:INPUT"WHICH TRACK WOULD YOU LIKE TO TRY";T:P=0:R=0:N=0
80 IFT<1THENT=1ELSEIFT>5THENT=5
90 ONTGOSUB1730,1540,570,850,1110
100 PRINT@952,"         ";:PRINT@941,"HORIZ MOVE";
110 INPUTM:PRINT@952,"         ";:PRINT@941,"VERT. MOVE";
120 INPUTV:PRINT@960,"
                ";:N=N+1:PRINT@812,N"SHOTS SO FAR";:REM 60 SPAC
ES
130 IFM<>99THEN160
140 CLS:INPUT"WOULD YOU LIKE TO PLAY AGAIN (Y/N)";A$
150 IFA$="Y"THEN30ELSEEND
160 IFM+V=P+R+1THEN200
170 IFM+V=P+R-1THEN200
180 PRINT@965,"THAT MOVE IS INVALID!";
190 PRINT@1006,"TRY AGAIN";:GOTO100
200 IFX+2*M<2THEN230ELSEIFX+2*M>125THEN230
210 IFY-V<1THEN230ELSEIFY-V>43THEN230
220 GOTO240
230 PRINT@965,"YOU ARE OUT OF BOUNDS!";:GOTO190
240 IFPOINT(X+2*M,Y-V)=-1THENPRINT@965,"YOU HAVE HIT SOMETHING!"
;:GOTO100
250 PRINT@877,"LAST MOVE";M;V;:P=M:R=V
260 X=X+2*M:Y=Y-V:SET(X,Y):SET(X+1,Y)
270 FORW=104TO112STEP2:FORZ=33TO35
280 IFPOINT(W,Z)=-1THEN300
290 NEXTZ,W:GOTO100
300 PRINT@965,"YOU HAVE MADE IT!  YOU TOOK"N"SHOTS. ";:INPUT"HIT
 <<ENTER>>";A$
310 GOTO140
320 CLS:PRINTTAB(20)"I N S T R U C T I O N S"
330 PRINT:PRINT"YOUR TASK IS TO MOVE AROUND THE COURSE":PRINT"BY
 SUPPLYING MOVES TO BE USED."
340 PRINT"EACH MOVE MUST DIFFER FROM YOUR PREVIOUS MOVE BY 1 AND
 ONLY 1":PRINT
350 PRINT"FOR EXAMPLE :  YOUR LAST MOVE WAS SAY 2,3...."
360 PRINT"YOUR NEXT MOVE COULD BE 1,3 OR 3,3 OR 2,2 OR 2,4."
370 PRINT:PRINT"POSITIVE MOVES MAKE YOU MOVE UP OR TO THE RIGHT"
380 PRINT"WHILE NEGATIVE MOVES ARE DOWN OR TO THE LEFT."
390 PRINT"YOU SELECT 1 OF 5 TRACKS PLUS THE OPTION OF OBSTACLES.
"
400 PRINT"THE NUMBER OF SHOTS AND YOUR LAST MOVE ARE SHOWN ON TH
E SCREEN."
410 PRINT"TO WIN, ALL YOU MUST DO IS LAND IN THE INDICATED AREA.
  SIMPLE!"
420 GOSUB440
430 CLS:PRINT@340,"IF YOU GET STUCK,":PRINT@400,"ENTER 99,99 AS
YOUR MOVE!":GOSUB440:RETURN
440 PRINT@980,"HIT <<ENTER>> ";:INPUTA$:RETURN
450 CLS:FORI=0TO127:SET(I,0):NEXT:  REM DRAW MAIN OUT LINE
460 FORI=1TO32:SET(126,I):SET(127,I):NEXT
470 FORI=125TO114STEP-1:SET(I,32):NEXT
480 FORI=103TO86STEP-1:SET(I,32):NEXT
490 FORI=33TO44:SET(86,I):SET(87,I):NEXT
500 FORI=85TO0STEP-1:SET(I,44):NEXT
510 FORI=43TO1STEP-1:SET(0,I):SET(1,I):NEXT
520 PRINT@749,"LAND >>     << HERE";
530 PRINT@813,"NO. OF MOVES";
540 PRINT@877,"START WITH 0,0";
550 RETURN
560 PRINT@965,"DO YOU WANT THE OPTIONAL OBSTACLES ?";:PRINT@941,
"";:INPUTA$:RETURN
570 GOSUB450:X=36:Y=37:SET(X,Y):SET(X+1,Y)
580 FORI=43TO29STEP-1:SET(40,I):SET(41,I):NEXT
590 FORI=39TO20STEP-1:SET(I,29):NEXT
600 FORI=28TO9STEP-1:SET(20,I):SET(21,I):NEXT
610 FORI=1TO20:SET(40,I):SET(41,I):NEXT
620 FORI=42TO81:SET(I,20):NEXT
630 FORI=21TO32:SET(64,I):SET(65,I):NEXT
640 FORI=60TO97:SET(I,9):NEXT
650 FORI=9TO31:SET(96,I):SET(97,I):NEXT
660 PRINT@782,CHR$(93);:PRINT@517,CHR$(91);:PRINT@80,CHR$(92);:P
RINT@528,CHR$(94);:PRINT@731,CHR$(92);:PRINT@805,CHR$(91);:PRINT
@357,CHR$(93);:PRINT@96,CHR$(94);:PRINT@121,CHR$(92);
670 GOSUB560:IFA$<>"Y"THENRETURN
680 FORI=10TO31:SET(I,9):NEXT
690 FORI=24TO36STEP4:SET(I,13):SET(I+1,13):NEXT
700 FORI=26TO34STEP4:SET(I,15):SET(I+1,15):NEXT
710 FORI=24TO36STEP4:SET(I,17):SET(I+1,17):NEXT
720 FORI=24TO26:FORJ=44TO53:SET(J,I):NEXTJ,I
730 FORI=42TO49:SET(I,33):NEXT:FORI=58TO65:SET(I,33):NEXT
740 FORI=76TO85:SET(I,32):NEXT
750 FORI=73TO52STEP-1:SET(I,14):NEXT
760 FORI=13TO6STEP-1:SET(52,I):SET(53,I):NEXT
770 FORI=54TO61:SET(I,6):NEXT
780 FORI=1TO4:SET(72,I):SET(73,I):NEXT
```

```
790 FORI=1TO4:SET(92,I):SET(93,I):NEXT
800 FORI=5TO8:SET(82,I):SET(83,I):NEXT
810 FORJ=100TO108STEP4:GOSUB840:NEXT
820 FORJ=114TO122STEP4:GOSUB840:NEXT
830 FORI=24TO26:FORJ=106TO117:SET(J,I):NEXTJ,I:RETURN
840 FORI=12TO14:SET(J,I):SET(J+1,I):NEXT:RETURN
850 GOSUB450:X=64:Y=11:SET(64,11):SET(65,11)
860 FORI=9TO15:SET(68,I):SET(69,I):NEXT
870 FORI=67TO44STEP-1:SET(I,15):NEXT
880 FORI=16TO43:SET(44,I):SET(45,I):NEXT
890 FORI=35TO8STEP-1:SET(28,I):SET(29,I):NEXT
900 FORI=30TO111:SET(I,8):NEXT
910 FORI=125TO82STEP-1:SET(I,17):NEXT
920 FORI=18TO23:SET(82,I):SET(83,I):NEXT
930 FORI=81TO64STEP-1:SET(I,23):NEXT
940 FORI=24TO35:SET(64,I):SET(65,I):NEXT
950 FORI=31TO25STEP-1:SET(94,I):SET(95,I):NEXT
960 FORI=96TO109:SET(I,25):NEXT
970 GOSUB560:IFA$<>"Y"THENRETURN
980 FORI=32TO40STEP4:FORJ=21TO27:SET(I,J):SET(I+1,J):NEXTJ,I
990 FORI=28TO38STEP2:FORJ=39TO41:SET(I,J-I/2+14):SET(I+1,J-I/2+1
4):NEXTJ,I
1000 FORI=4TO24STEP4:FORJ=18TO34STEP2:SET(I,J):SET(I+1,J):NEXTJ,
I
1010 FORI=14TO27:SET(I,15):SET(I,8):NEXT
1020 FORI=9TO14:SET(14,I):SET(15,I):NEXT
1030 PRINT@201,"KEEP";:PRINT@265,"OUT";
1040 FORI=42TO79:SET(I,1):SET(I,2):SET(I,6):SET(I,7):NEXT
1050 FORI=52TO69:SET(I,3):SET(I,5):NEXT
1060 FORI=86TO99:FORJ=3TO5:SET(I,J):NEXTJ,I
1070 FORI=76TO120STEP4:FORJ=11TO14:SET(I,J):SET(I+1,J):NEXTJ,I
1080 FORI=21TO38:SET(56,I):SET(57,I):NEXT
1090 FORI=96TO101:FORJ=26TO31:SET(I,J):NEXTJ,I
1100 FORI=92TO116STEP4:FORJ=19TO23STEP2:SET(I,J):SET(I+1,J):NEXT
J,I:RETURN
1110 GOSUB450:X=46:Y=19:SET(46,19):SET(47,19)
1120 FORI=54TO40STEP-2:SET(I,45-I/2):SET(I+1,45-I/2):NEXT
1130 FORI=26TO38:SET(40,I):SET(41,I):NEXT
1140 FORI=39TO10STEP-1:SET(I,38):NEXT
1150 FORI=37TO6STEP-1:SET(10,I):SET(11,I):NEXT
1160 FORI=12TO35:SET(I,6):NEXT
1170 FORI=7TO11:SET(34,I):SET(35,I):NEXT
1180 FORI=36TO81:SET(I,11):NEXT
1190 FORI=82TO90STEP2:SET(I,51-I/2):SET(I+1,51-I/2):NEXT
1200 FORI=90TO115:SET(I,6):NEXT
1210 FORI=6TO25:SET(114,I):SET(115,I):NEXT
1220 FORI=115TO102STEP-1:SET(I,25):NEXT
1230 FORI=25TO32:SET(102,I):SET(103,I):NEXT
1240 FORI=12TO32:SET(22,I):SET(23,I):NEXT
1250 FORI=23TO107:SET(I,17):NEXT
1260 FORI=94TO70STEP-2:SET(I,65-I/2):SET(I+1,65-I/2):NEXT
1270 FORI=31TO37:SET(70,I):SET(71,I):NEXT
1280 FORI=68TO54STEP-2:SET(I,56-I/2):SET(I+1,56-I/2):NEXT
1290 FORI=30TO43:SET(54,I):SET(55,I):NEXT
1300 GOSUB560:IFA$<>"Y"THENRETURN
1310 FORI=30TO41:FORJ=24TO26:SET(I,J):NEXTJ,I
1320 FORI=24TO32:SET(16,I):SET(17,I):NEXT
1330 FORI=14TO18STEP4:FORJ=12TO20:SET(I,J):SET(I+1,J):NEXTJ,I
1340 FORI=14TO16:SET(44,I):SET(45,I):SET(64,I):SET(65,I):NEXT
1350 FORI=12TO14:SET(54,I):SET(55,I):SET(74,I):SET(75,I):NEXT
1360 FORI=10TO17:SET(96,I):SET(97,I):NEXT
1370 FORI=21TO27:SET(90,I):SET(91,I):NEXT
1380 FORI=78TO89:SET(I,27):NEXT
1390 FORI=74TO82STEP4:FORJ=32TO36STEP2:SET(I,J):SET(I+1,J):NEXTJ
,I
1400 FORI=33TO37STEP2:SET(76,I):SET(77,I):SET(80,I):SET(81,I):NE
XT
1410 FORI=30TO41:SET(62,I):SET(63,I):NEXT
1420 FORI=64TO73:SET(I,41):NEXT
1430 FORI=58TO79:SET(I,20):NEXT
1440 FORI=4TO17:SET(I,41):NEXT
1450 FORI=36TO40:SET(4,I):SET(5,I):NEXT
1460 FORI=27TO32:SET(6,I):SET(7,I):NEXT
1470 FORI=18TO23:SET(4,I):SET(5,I):NEXT
1480 FORI=9TO14:SET(6,I):SET(7,I):NEXT
1490 FORI=26TO33:SET(I,1):SET(I,3):SET(I,5):NEXT
1500 FORI=40TO80STEP4:FORJ=2TO8STEP2:SET(I,J):SET(I+1,J):NEXTJ,I
1510 FORI=96TO103:SET(I,1):SET(I,3):SET(I,5):NEXT
1520 FORI=116TO123:FORJ=27TO30:SET(I,J):NEXTJ,I
1530 RETURN
1540 GOSUB450:X=80:Y=38:SET(80,38):SET(81,38)
1550 FORI=85TO26STEP-1:SET(I,32):NEXT
1560 FORI=2TO27:SET(I,21):NEXT
1570 FORI=31TO11STEP-1:SET(52,I):SET(53,I):NEXT
1580 FORI=28TO105:SET(I,10):NEXT
1590 FORI=78TO125:SET(I,20):NEXT
1600 PRINT@795,CHR$(93);:PRINT@583,CHR$(94);:PRINT@340,CHR$(93);
:PRINT@72,CHR$(94);:PRINT@123,CHR$(92);:PRINT@366,CHR$(93);:PRIN
T@566,CHR$(93);
1610 GOSUB560:IFA$<>"Y"THENRETURN
1620 FORI=14TO30STEP4:FORJ=36TO41:SET(I,J):SET(I+1,J):NEXTJ,I
1630 FORI=10TO42STEP2:SET(I,I/2+4):SET(I+1,I/2+4):NEXT
1640 FORI=4TO9:SET(36,I):SET(37,I):NEXT
1650 FORI=1TO6:SET(54,I):SET(55,I):NEXT
1660 FORI=4TO9:SET(72,I):SET(73,I):NEXT
1670 FORI=1TO6:SET(90,I):SET(91,I):NEXT
1680 FORI=106TO116STEP2:SET(I,I/2-43):SET(I+1,I/2-43):NEXT
1690 FORI=64TO69:FORJ=15TO17:SET(I,J):NEXTJ,I
1700 FORI=76TO81:FORJ=15TO17:SET(I,J):NEXTJ,I
1710 FORI=64TO69:FORJ=24TO26:SET(I,J):NEXTJ,I
1720 FORI=76TO81:FORJ=24TO26:SET(I,J):NEXTJ,I:RETURN
1730 GOSUB450:X=4:Y=6:SET(4,6):SET(5,6)
1740 FORI=2TO27:SET(I,14):NEXT
1750 FORI=1TO29:SET(54,I):SET(55,I):NEXT
1760 FORI=53TO26STEP-1:SET(I,29):NEXT
1770 FORI=31TO15STEP-1:SET(86,I):SET(87,I):NEXT
1780 PRINT@139,CHR$(94);:PRINT@279,CHR$(92);:PRINT@520,CHR$(92);
:PRINT@792,CHR$(94);:PRINT@611,CHR$(91);:PRINT@182,CHR$(92);
```

```
1790 GOSUB560:IFA$<>"Y"THENRETURN
1800 FORI=9TO20:SET(36,I):SET(37,I):NEXT
1810 FORI=12TO32STEP4:FORJ=33TO39STEP2:SET(I,J):SET(I+1,J):NEXTJ
,I
1820 FORI=58TO82STEP8:FORJ=18TO26:SET(I,J):SET(I+1,J):NEXTJ,I
1830 FORI=68TO81:SET(I,10):NEXT
1840 FORI=7TO13:SET(74,I):SET(75,I):NEXT
1850 FORI=94TO122STEP2:SET(I,I/2-35):SET(I+1,I/2-35):SET(I,73-I/
2):SET(I+1,73-I/2):NEXT
1860 RETURN
```

****** LII/4K to 48K m/l CHECKSUM ******

TRS-80/SYSTEM-80

```
00010 ;        * * * * * CHECKSUM * * * * *
00020 ;        * * * BY  B. N. BRIGGS * * *
00030 ;        * * * * NAPIER, N.Z. * * * *
00040 ;        * * * * JULY,    1982 * * * *
00050 ;
00060 ;        * LEVEL 2 -- ALL MEM SIZES *
00070 ;
00080 ;
00250          ORG     81E2H           ;SYSTEM ENTRY POINT
00260 SYST     JP      ENTRY           ;GOTO INITIALISATION
00270 MMORY    DEFW    0               ;MEMORY BOUNDRY SO ALLOW
00280                                  ;     A COUPLE OF BYTES
00290 START    PUSH    HL              ;SAVE ERROR POINT
00300          EX      AF,AF'          ;SAVE ALL
00310          EXX                     ;       REGISTERS
00320          POP     HL              ;GET ERROR
00330          INC     HL              ;    POINT BACK
00340 TABLE1   LD      DE,TABLE        ;GET WORD TO CHECK
00350          LD      B,5             ;NO OF LETTERS
00360 LOOK     LD      A,(DE)          ;GET LETTER
00370          CP      (HL)            ;SAME AS WORD ENTERED?
00380          JR      NZ,GOHOME       ;NO! THEN GO HOME
00390          INC     HL              ;NEXT LETTER
00400          INC     DE              ; TO COMPARE
00410          DJNZ    LOOK            ;HO HUM CHECK NEXT LETTER
00420                                  ;AND DO IT 5 TIMES
00430          JR      COMENC          ;ALL CORRECT SO CARRY ON
00440 GOHOME   EX      AF,AF'          ;NOT CORRECT SO RESTORE
00450          EXX                     ;    ALL REGISTERS AND
00460          RET                     ;          RETURN
00470 TABLE    DEFM    'CHECK'         ;WORD TO COMPARE
00480 COMENC   POP     HL              ;GOOD HOUSEKEEPING -
00490                                  ;REMOVE RET OFF STACK
00500          LD      HL,3F40H        ;SHIFT THE CURSOR DOWN
00510          LD      (4020H),HL      ; TO THE BOTTOM OF SCREEN
00520 MES1     LD      HL,MESG1        ;GET THE PROG LENGTH MESG
00530          CALL    28A7H           ;  AND PRINT IT
00540          LD      HL,(40F9H)      ;GET THE END OF PROGRAM
00550          LD      DE,(40A4H)      ;    AND THE START !
00560          PUSH    HL              ;SAVE THE END & THE START
00570          PUSH    DE              ;        TIL LATER
00580          DEC     HL              ;TAKE THE END DOWN A
00590          DEC     HL              ;     PEG OR TWO
00600          SBC     HL,DE           ;GET THE PROGRAM LENGTH
00610          CALL    0FAFH           ;    AND PRINT IT !
00620          POP     HL              ;BRING BACK START OF
00630                                  ;    BASIC PROG INTO 'HL'
00640          LD      E,(HL)          ;GET LSB OF START OF 2ND
00650          INC     HL              ;  LINE AND ADVANCE ONE
00660          LD      D,(HL)          ;GET MSB OF 2ND LINE
00670          POP     HL              ;BRING BACK THE END
00680          LD      A,D             ;CHECK IF THERE
00690          OR      E               ;         IS A SECOND LINE
00700          JR      Z,BASIC         ;NO! GO BACK TO BASIC
00710          SBC     HL,DE           ;NUMBER OF BYTES TO COUNT
00720          PUSH    DE              ;SAVE START OF 2ND LINE
00730          PUSH    HL              ;      AND NUMBER OF BYTES
00740 MES2     LD      HL,MESG2        ;GET CHECKSUM MESSAGE
00750          CALL    28A7H           ;  AND PRINT IT
00760          POP     BC              ;NUMBER OF BYTES INTO BC
00770          POP     DE              ;BRING BACK 2ND LINE
00780          LD      HL,0000H        ;ZERO HL
00790 LOOP     LD      A,(DE)          ;GET THE VALUE OF BYTE
00800          PUSH    DE              ;SAVE YOUR POSITION
00810          LD      D,00H           ;ZERO 'D'
00820          LD      E,A             ;LOAD 'A' INTO 'DE'
00830          ADD     HL,DE           ;ADD IT TO 'HL'
00840          POP     DE              ;BRING BACK YOUR POSITION
00850          INC     DE              ;GO TO NEXT POSIE'
00860          DEC     BC              ;ONE LESS BYTE TO COUNT
00870          LD      A,B             ;CHECK IF
00880          OR      C               ;         'BC' = ZERO
00890          JR      NZ,LOOP         ;NO? GET SOME MORE. YES?
00900          CALL    0FAFH           ;THEN PRINT WHATS IN 'HL'
00910 BASIC    JP      06CCH           ;& GO BACK TO BASIC
00920 MESG1    DEFM    'PROGRAM LENGTH = '
00930          DEFB    0               ;END OF MESSAGE
00940 MESG2    DEFB    0C5H            ;ADD A BIT OF SPACE
00950          DEFM    'CHECKSUM = '
00960          DEFB    0               ;END OF MESSAGE
00970 LAST     DEFB    0               ;LAST BYTE TO BE MOVED
00980 ; INITIALISATION ROUTINE
00990 OFFSET   EQU     LAST-MMORY
01000          ORG     828AH           ;LEAVE A GAP FOR STACK
01010 ENTRY    LD      SP,4288H        ;MAKE SURE ITS THERE !
01020          LD      A,0C9H          ;ALTER SYSTEM ENTRY POINT
01030          LD      (SYST),A        ;       BACK TO RETURN
```

```
01040          LD       A,0C3H          ;PATCH JUMP INTO BASIC
01050          LD       (41B2H),A       ;       INPUT
01060          LD       HL,(40B1H)      ;GET PRESENT MEM BOUNDRY
01070          LD       DE,1-OFFSET     ;PLUS OFFSET = START OF
01080          ADD      HL,DE           ;      PROGRAM FOR PATCH
01090          LD       (41B3H),HL      ;BASIC INPUT POINT
01100          PUSH     HL              ;SAVE START
01110          LD       DE,MMORY+1      ;GET START OF PROGRAM
01120          SBC      HL,DE           ;OFFSET - NEW PROG POSIE
01130          EX       DE,HL           ;PUT IT INTO 'DE'
01140          LD       HL,MESG1        ;GET FIRST MESSAGE POSIE'
01150          ADD      HL,DE           ;ADD OFFSET
01160          LD       (MES1+1),HL     ;AND PUT IT BACK
01170          LD       HL,MESG2        ;DO THE SAME WITH THE
01180          ADD      HL,DE           ;      SECOND MESSAGE
01190          LD       (MES2+1),HL     ;      POSITION
01200          LD       HL,TABLE        ;START OF WORD 'CHECK'
01210          ADD      HL,DE           ;ADD OFFSET
01220          LD       (TABLE1+1),HL   ;AND PUT IT BACK
01230          POP      HL              ;BRING BACK START OF PROG
01240          DEC      HL              ;ALLOW A COUPLE
01250          DEC      HL              ;OF BYTES
01260          LD       (40B1H),HL      ;AND CLOSE THE MEM DOOR
01270          PUSH     HL              ;SAVE MEM BOUNDRY
01280          LD       DE,0-50         ;OFFSET FOR STRING SPACE
01290          ADD      HL,DE           ;ADD IT IN
01300          LD       (40A0H),HL      ;AND ENTER IT
01310          LD       HL,MMORY        ;WHERE WE IS
01320          POP      DE              ;WHERE WE WANNA BE
01330          LD       BC,LAST-MMORY   ;NO OF BYTES TO MOVE
01340          LDIR                     ;DEEP BREATH & MOVE IT
01350          XOR      A               ;ZERO 'A'
01360          CALL     1E7AH           ;CALL THE CLEAR ROUTINE
01370          JP       06CCH           ;AND GO BACK TO BASIC
01380          END      SYST
```

```
CHECKSUM DUMP
START    END      ENTRY
41E2     42DA     41E2
```

```
41E2:  C3 8A 42 00 00 E5 08 D9 E1 23 11 FE 41 06 05 1A
41F2:  BE 20 06 23 13 10 F8 18 08 08 D9 C9 43 48 45 43
4202:  4B E1 21 40 3F 22 20 40 21 4B 42 CD A7 28 2A F9
4212:  40 ED 5B A4 40 E5 D5 2B 2B ED 52 CD AF 0F E1 5E
4222:  23 56 E1 7A B3 28 1F ED 52 D5 E5 21 5D 42 CD A7
4232:  28 C1 D1 21 00 00 1A D5 16 00 5F 19 D1 13 0B 78
4242:  B1 20 F3 CD AF 0F C3 CC 06 50 52 4F 47 52 41 4D
4252:  20 4C 45 4E 47 54 48 20 3D 20 00 C5 43 48 45 43
4262:  4B 53 55 4D 20 3D 20 00 00 00 00 00 00 00 00 00
4272:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4282:  00 00 00 00 00 00 00 00 31 88 42 3E C9 32 E2 41
4292:  3E C3 32 B2 41 2A B1 40 11 7C FF 19 22 B3 41 E5
42A2:  11 E6 41 ED 52 EB 21 4B 42 19 22 0B 42 21 5D 42
42B2:  19 22 2E 42 21 FE 41 19 22 ED 41 E1 2B 2B 22 B1
42C2:  40 E5 11 CE FF 19 22 A0 40 21 E5 41 D1 01 85 00
42D2:  ED B0 AF CD 7A 1E C3 CC 06
```

****** LI/4K MATRIX MANIPULATION ******

TRS-80/SYSTEM-80

```
10  REM***** MATRIX MANIPULATION : (C) 1980 BY MARTIN DOWNEY **
20  Y=0:I=1:CLS:P.T.(12);"MATRIX MANIPULATION (BY MARTIN DOWNEY)
";P.
100  P.:P.Y;"MATRICES ALREADY DEFINED":P."MATRIX INSTRUCTIONS:"
120  P."PRINT=1 : DEFINE=2 : MULT=3 : ADD/SUB=4 : INVERSE=5 : DE
LETE=6"
130  IN.Z:CLS:ONZGOS.200,400,800,1000,2000,5000
140  G.100
200  IFY=ORET.
210  W=0:F.X=1TOY:P."MATRIX";X
220  W=W+2:M=A(W-1):N=A(W)
230  F.R=1TOM:F.C=1TON:W=W+1:P.T.((C-1)*6),A(W);
240  N.C:P.:N.R:IN."PRESS (ENTER) TO CONTINUE";A$:IFZ<>1RET.
250  N.X:RET.
400  CLS:P."MATRIX";Y+1
410  IN."NUMBER OF ROWS & COLUMNS: R,C";M,N:A(I)=M:A(I+1)=N
420  IFM./4-1<I+M*N+26.6000
430  Y=Y+1:I=I+2:P.AT(128);"ENTER ONE ELEMENT AFTER EACH  '?'"
440  F.R=1TOM:P.AT(128+R*64);"ROW";R;:F.C=1TON:P.AT(130+R*64+C*6
);
450  IN.A(I):I=I+1:N.C:N.R:RET.
800  IN."FIRST MATRIX'S NUMBER";A:IF(A<1)+(A>Y)RET.
810  IN."MULTIPLIED BY: CONSTANT=1  OR  MATRIX=2";V:IFV=1T.900
815  IN."SECOND MATRIX'S NUMBER";B:IF(B<1)+(B>Y)RET.
835  GOS.4000:A(I-1)=P:IFN<>OP."NOT POSSIBLE":I=I-2:RET.
840  IFM./4-1<I+M*P+26.6000
850  F.J=0TO(M-1):F.H=0TO(P-1):A(I)=0:F.K=0TO(N-1)
860  A(I)=A(I)+A(L+J*N+K)*A(Q+K*P+H):N.K
870  I=I+1:N.H:N.J:Y=Y+1
880  P."MATRIX";A;"TIMES MATRIX";B;"GIVES MATRIX";Y:G.220
900  IN."CONSTANT OF MULTIPLICATION";K:GOS.4000
940  F.J=1TO(M*N):A(I)=A(L+J-1)*K:I=I+1:N.J
950  Y=Y+1:P."MATRIX";A;"TIMES";K;"GIVES MATRIX";Y:G.220
1000  B$="PLUS (":IN."FIRST MATRIX'S NUMBER";A:IF(A<1)+(A>Y)RET.
1010  IN."MULTIPLE OF ADDITION. (-1 GIVES MINUS , +1 GIVES PLUS)
";V
1020  IFV<0B$="MINUS ("
1030  IN."SECOND MATRIX'S NUMBER";B:IF(B<1)+(B>Y)RET.
1060  GOS.4000:IF(M<>O)+(N<>P)P."NOT POSSIBLE":I=I-2:RET.
1070  IFM./4-1<I+M*N+26.6000
```

```
1080  F.J=0TO(M*N-1):A(I)=A(L+J)+V*A(Q+J):I=I+1:N.J:Y=Y+1
1090  P."MATRIX";A;B$;ABS(V);"TIMES MATRIX";B;") GIVES MATRIX";Y
:G.220
2000  IN."MATRIX NUMBER";A:IF(A<1)+(A>Y)RET.
2010  GOS.4000:L=L-1:I=I-1:IFN<>MP."NOT SQUARE":I=I-1:RET.
2012  IFM./4-1<M*N+I+2G.6000
2014  F.J=1TON:F.K=1TON:IFK=JA((J-1)*N+K+I)=1:G.2018
2016  A((J-1)*N+K+I)=0
2018  N.K:N.J
2020  F.Q=1TO(N-1):F.J=QTO(N-1):U=1:G.3000
2025  IFA((Q-1)*N+Q+L)=0T.2200
2030  R=A(J*N+Q+L)/A((Q-1)*N+Q+L):F.K=1TON
2040  A(J*N+K+L)=A(J*N+K+L)-R*A((Q-1)*N+K+L)
2050  A(J*N+K+I)=A(J*N+K+I)-R*A((Q-1)*N+K+I):N.K:N.J:N.Q
2060  F.Q=1TO(N-1):F.J=(Q+1)TON:U=2:G.3000
2070  R=A((Q-1)*N+J+L)/A((J-1)*N+J+L):F.K=1TON
2080  A((Q-1)*N+K+L)=A((Q-1)*N+K+L)-R*A((J-1)*N+K+L)
2090  A((Q-1)*N+K+I)=A((Q-1)*N+K+I)-R*A((J-1)*N+K+I)
2100  N.K:N.J:N.Q
2110  F.J=0TO(N-1):F.K=1TON:A(J*N+K+I)=A(J*N+K+I)/A(J*N+J+1+L):N
.K:N.J
2180  I=I+N*N+1:Y=Y+1:GOS.5010:W=I-N*N-3
2190  P."MATRIX";A;"HAS BEEN DELETED. IT'S INVERSE IS MATRIX";Y:
G.220
2200  F.O=JTO(N-1):IFA(O*N+Q+L)=0T.2300
2210  F.P=QTON:A((Q-1)*N+P+L)=A((Q-1)*N+P+L)+A(O*N+P+L)
2220  A((Q-1)*N+P+I)=A((Q-1)*N+P+I)+A(O*N+P+I):N.P:G.2030
2300  N.O
2310  P."MATRIX";A;"DELETED. SINGULAR.":GOS.5010:I=I-1:RET.
3000  F.P=1TON:T=0:S=0:F.O=1TON:IFA((P-1)*N+O+L)=0T=T+1
3005  IFA((O-1)*N+P+L)=0S=S+1
3010  N.O:IF(S=N)+(T=N)G.2310
3020  N.P:ONUG.2025,2070
4000  W=1:F.J=1TOY:IFJ=AM=A(W):N=A(W+1):L=W+2
4010  IFJ=BO=A(W):P=A(W+1):Q=W+2
4020  W=W+2+A(W)*A(W+1):N.J:A(I)=M:A(I+1)=N:W=I-1:I=I+2:RET.
5000  IN."MATRIX NUMBER";A:IF(A<1)+(A>Y)RET.
5010  GOS.4000:Y=Y-1:I=I-M*N-4
5020  F.J=(L-2)TO(I-1):A(J)=A(J+M*N+2):N.J:RET.
6000  P."INSUFFICIENT MEMORY. TRY DELETE.":RET.
```

****** LII/4K INPUT DEMONSTRATION ******

TRS-80/SYSTEM-80

```
10 CLS
20 'AN ALTERNATIVE TO THE INPUT COMMAND (LEVEL 2,4K/16K)
30 '================================
40 'BY KEN SHILLITO,10 MILTON ST.,CHATSWOOD 2067
50 DEFSTR A-B            :REM A$ WILL HOLD INPUT STRING
60 DEFINT I-M            :REM VARIABLES USED BY KEY SUBROUTINE
70 '
80 'EXTRACT FROM BODY OF PROGRAM -SUPPOSE AN INPUT IS REQUIRED
   TO GIVE AN INTEGER VALUE FROM 1 TO 9999 TO X
90 LET I=4               :REM MAXIMUM INPUT LENGTH
100 LET J=50              :REM SCREEN ADDRESS FOR ECHO
110 PRINT@0,"TYPE AN INTEGER FROM 0 TO 9999"     :REM PROMPT
120 GOSUB160            :REM GET INPUT
130 LET X=VAL(A)          :REM GIVE X ITS VALUE,AND CONTINUE
140 PRINT X:END           :REM SEE IF IT WORKED
150 '
160 'THE KEY SUBROUTINE WHICH DOES THE WORK
170 LET A=""              :REM A WILL HOLD THE INPUT
180 PRINT@J,STRING$(I," ")
190 LET B=STR$(31)        :REM MAXIMUM CONTROL VARIABLE
200 LET L=1               :REM CONSTANT
210 LETA=A+INKEY$:PRINT@J,A:IFLEN(A)<IANDRIGHT$(A,L)>BTHEN210
220 IF LEN(A)=0 THEN210
230 LET M=ASC(RIGHT$(A,L))      :REM TEST FOR CONTROL CHRS
240 IF M>31 RETURN        :REM NOT A CONTROL CHARACTER
250 IF M=24 THEN160   :REM ERASE LINE
260 LET A=LEFT$(A,LEN(A)-1)     :REM REMOVE CONTROL CHR
270 IF M=13 RETURN        :REM ENTER/NEW LINE KEY
280 IF LEN(A)>0 LET A=LEFT$(A,LEN(A)-1)   :REM BACKSPACE
290 GOTO210             :CONTINUE WITH INPUT
```

****** LII/4K FLEXITIME ******

TRS-80/SYSTEM-80

```
10 CLEAR200:CLS:PRINT CHR$(23);"
  ** FLEXITIME ADDER **":PRINT"
  COPYRIGHT DAVID GRIGG
        1980":FORZ=1TO1000:NEXTZ
20 CLS:ST=7*60+21 'STANDARD DAY
30 PRINT "ENTER TIMES IN THE FORMAT: 7.21 (= 7 HR 21 MIN)
40 PRINT"
yOU MAY ENTER TIMES IN EITHER 12 OR 24-HOUR FORMAT, OR EVEN
MIXED, EG. START 13.00, END 4.00, SO LONG AS THESE TIMES ARE
RATIONAL FOR A NORMAL WORKING DAY.
"
50 PRINT"YOUR TOTAL OF FLEXITIME CREDIT OR DEBIT AT THE BEGINNIN
G OF
THE CURRENT WORKING PERIOD SHOULD BE ENTERED NOW.
WHILE THE PROGRAM IS RUNNING IT WILL KEEP TRACK OF THE TOTAL.
"
60 PRINT"PROGRESSIVE TOTAL (+ OR -)";:GOSUB290
70 CLS:PR=NQ:PRINT@512,STRING$(64,"-")
```

```
80 GOSUB360:PRINT"MORNING START=";:GOSUB300:SM=NQ:PRINT@704,BL$:
PRINT@64,"";
90 PRINT"MORNING FINISH=";:GOSUB300:FM=NQ:MO=FM-SM
100 PRINT"AFTERNOON START=";:GOSUB300:SA=NQ
110 PRINT"AFTERNOON FINISH=";:GOSUB300:FA=NQ:AF=FA-SA
120 DAY=MO+AF:MN=DAY:GOSUB220
130 PRINT@640,"HOURS WORKED";TAB(15);"FLEXITIME";TAB(30);"PROGRE
SSIVE TOTAL"
140 HW=HM::PRINTUSING"+####.##";HW;:PRINT"        ";
150 FL=DAY-ST:MN=FL:GOSUB220
160 FH=HM::PRINTUSING"+####.##";FH;:PRINT"        ";
170 PR=PR+FL:MN=PR:GOSUB220
180 PP=HM:PRINTUSING"+####.##";PP;:PRINT" ";
190 IF ABS(PP)>7.21 THEN
        IF PP<0 PRINT STRING$(INT(ABS(PP))-6,"!")
        ELSE PRINT STRING$(INT(ABS(PP))-6,"$")
200 GOTO 80
210 END
220 'REASSEMBLE INTO HOURS AND MINUTES
230 S=SGN(MN):MN=ABS(MN)
240 HR=INT(MN/60)
250 MM=INT(MN-HR*60+.01):IF MM=60  THEN HR=HR+1:MM=0 ELSE MM=MM/
100
260 HM=S*(HR+MM)
270 RETURN
280 'RATIONALISE FOR NON TIME-OF-DAY DATA
290 INPUT Q:GOTO320
300 ' RATIONALISE HOURS AND MINUTES INTO ALL MINUTES
310 INPUT Q: IF Q<7 THEN Q=Q+12
320 HQ=FIX(Q)*60
330 MQ=(Q-FIX(Q))*100
340 NQ=HQ+MQ
350 RETURN
360 BL$=STRING$(64," ")
370 PRINT@0,BL$:PRINT@64,BL$:PRINT@128,BL$:PRINT@192,BL$:PRINT@0
,"";
380 RETURN
```

****** LII/4K CODE BREAKER ******

TRS-80/SYSTEM-80

```
10 CLS:PRINTCHR$(23):PRINT@334,"CODEBREAKER
          BY
      DAVID GRIGG":FORZ=1TO1000:NEXTZ:CLS
20 CLEAR2000:DEFINTA-Z
30 DIM KE(26),A$(7),C$(7),T(26),S(20),SS(20)
40 FOR N=1TO26:KEY(N)=-19:NEXTN:KEY(0)=-32
50 FORQ=1TO10:READ D(Q):NEXT Q
60 CLS:PRINT"ENTER CODE TEXT, HITTING 'ENTER' AT END OF EACH LIN
E
END TEXT WITH '#' ON A NEW LINE"
70 FORN=1TO6:INPUTA$(N):IFA$(N)="#"THENGOTO80:ELSENEXTN
80 L=N-1:IFL>4THENSP=0ELSESP=1
90 INPUT"MOST COMMON LETTER SUBSTITUTION";Q$:IFLEFT$(Q$,1)="Y"TH
EN GOSUB 320
100 GOSUB 110:GOSUB 240:GOTO100
110 CLS:FORN=1TOL:C$(N)=""
120 : FOR M=1TOLEN(A$(N))
130 :   K=ASC(MID$(A$(N),M,1))-64:IFK<0THENC$(N)=C$(N)+CHR$(K+64
):GOTO170
140 :   IF K>32 THEN K=K-32
150 :   J=KEY(K)
160 :   C$(N)=C$(N)+CHR$(J+64)
170 : NEXT M
180 : GOSUB210
190 NEXT N
200 RETURN
210 'DISPLAY
220 PRINT A$(N):PRINT C$(N):IFSP=1THEN:PRINT
230 RETURN
240 PRINT STRING$(63,"-")
250 FOR N=1TO26:PRINT CHR$(N+64);:NEXT N:PRINT
260 FOR N=1TO26:PRINT CHR$(KEY(N)+64);:NEXT:PRINT"  ";
270 INPUT"SUBSTITUTE (EG. A/F)";B$
280 K=ASC(LEFT$(B$,1))-64
290 J=ASC(RIGHT$(B$,1))-64
300 KEY(K)=J:FORN=1TO26:IFKEY(N)=JANDN<>KTHENKEY(N)=-19:NEXTN:EL
SENEXTN
310 RETURN
320 FOR N=1TOL
330 : FOR M=1TOLEN(A$(N))
340 :   K=ASC(MID$(A$(N),M,1))-64:IF K<0 OR K>26 THEN K=0
350 :   T(K)=T(K)+1
360 : NEXT M
370 NEXT N
380 MAX=32000:Q=1
390 LMAX=MAX:MAX=0
400 FOR N= 1 TO 26
410 : IF T(N)>MAX AND T(N)<LMAX THEN MAX=T(N)
420 NEXT N
430 FOR N=1 TO 26
440 : IF T(N)=MAX THEN S(Q)=N:SS(Q)=MAX:Q=Q+1
450 NEXT N
460 IF Q<11 THEN 390
470 CLS:PRINT"MOST FREQUENT CODE LETTERS AND SUBSTITUTES
(NO.)","CODE","FREQ","SUB":FORQ=1TO10:PRINT"(";Q;")",CHR$(S(Q)+6
4),SS(Q),CHR$(D(Q)+64):NEXTQ:INPUT"
SUBSTITUTE DOWN TO NUMBER WHAT";B
480 FORQ=1TOB:IFS(Q)<=0THEN490ELSEKEY(S(Q))=D(Q)
490 NEXTQ
500 RETURN
510 DATA 5,20,1,15,9,14,19,8,18,4
```

***** NEXT MONTH'S ISSUE *****

Next month's issue will contain at least the following programs plus the usual features and articles. An (80) after a program title indicates that the program will be for TRS-80 Model 1/3 or System 80/Video Genie computers. (Colour) indicates that the program will be for the TRS-80 Colour Computer and the Hitachi Peach.

** ANAGRAMS LII/16K (80) **

If you like anagrams, then this is the program for you. It has several levels of difficulty and even if you are typing in this program from the magazine, you still won't know what the answers are as they are coded in the program listing.

** GOLF LII/16K (80) **

This Golf game has a full 18 holes and even a putting practice section. The speed of the graphics has been enhanced by the use of inbuilt machine code. When you have finished playing this game, it even invites you to go to the nineteenth hole.

** SINKING THE ENEMY NAVY (COLOUR) **

This is a computer version of that old favourite board game of Battleships, where the player has to supply the vectors on a grid where he thinks a ship might be. This version has the different types of ships in different colours and also has HIT and MISS sounds.

** E=MC2 LII/4K (80) **

This program actually relates to the relativistic increase in mass, as predicted by Einstein's Theory of Relativity, as its velocity approaches the velocity of light. You enter the mass (in kilograms) of a body at rest and then enter its speed and this program calculates what happens to its mass.

** SOUTH AUSTRALIAN HORSE PERFORMANCE GUIDE ** LII/16K (80)

We don't promise that you will always pick winners with this program, but it sure will help. This tape based program is so massive it has to be loaded into your machine in three parts.

** MASTERMIND (COLOUR) **

In this all new version of Mastermind you can choose between having numbers or letters hidden away and you can choose how many numbers or letters you wish to have in each game. So you can make this Mastermind as easy or as hard as you like.

APPLICATION FOR PUBLICATION OF A PROGRAM IN MICRO-80

Date .....

To **MICRO-80** SOFTWARE DEPT. PO BOX 145 MORPHETT VALE SA 5162

Please consider the enclosed program for . . .

Tick where appropriate

(i) Publication in MICRO-80 .....

(ii) Publication on disk or cassette only .....

(iii) Both .....

Name .....

Address .....

.....

.....Postcode .....

*** * * CHECK LIST * * ***

Please ensure that the cassette or disk is clearly marked with your name and address, program name(s), Memory size, Level I, II, System 1 or 2, Edtasm, System, etc. The use of REM statements with your name and address is suggested, in case the program becomes separated from the accompanying literature.

Ensure that you supply adequate instructions, notes on what the program does and how it does it, etc.

For system tapes, the start, end, and entry points, etc.

The changes or improvements that you think may improve it.

Please package securely -- padabags are suggested -- and enclose stamps or postage if you want your cassette or disk returned.

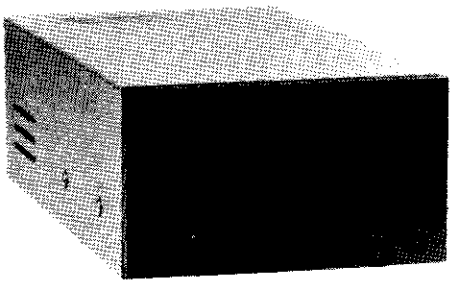
***** CASSETTE/DISK EDITION INDEX *****

The cassette edition of MICRO-80 contains all the software listed each month, on cassette. The cassette also contains the source code for machine language programs which may not have been printed due to space restrictions. All programs are recorded twice. Level I programs can only be loaded into a Level I TRS-80 if the Level I in Level 2 program from the MICRO-80 Software Library - Vol. 1 is first loaded into your Level 2 TRS-80 or System 80/Video Genie. Note: System 80/Video Genie computers have had different tape-counters fitted at different times. The approximate start positions shown are correct for the very early System 80 without the volume control or level meter. They are probably incorrect for later machines. The rates for a cassette subscription are printed on the inside front cover of each issue of the magazine.

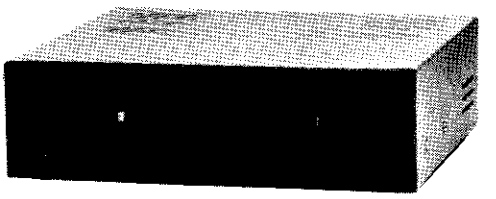
The disk edition contains all those programs which can be executed from disk, including Level I programs. Level I disk programs are saved in the NEWDOS format. Users require the Level I/CMD utility supplied with NEWDOS+ or NEWDOS 80 version 1.0 to run them.

SIDE 1	TYPE	I.D.	DISK FILESPEC	APPROX. START POSITION CTR-41	CTR-80	SYSTEM 80
INPUT DEMONSTRATION	LII/4K	I	INPUT/BAS	18	10	5
" "	"	"	"	36	20	8
MOVE BY 1'S	LII/16K	M	MOVE/BAS	54	30	13
" "	"	"	"	124	69	35
FLEXITIME	LII/4K	F	FLEXI/BAS	190	106	60
"	"	"	"	207	116	66
CODE BREAKER	LII/4K	B	BREAKER/BAS	225	126	72
" "	"	"	"	241	135	77
CHECKSUM	SYSTEM	CHECK	CHECKSUM/CMD	257	144	83
"	"	"	"	265	148	85
CHECKSUM	EDTASM	CHECK	CHECKSUM/EDT	273	153	90
"	"	"	"	316	177	107
MOVIE UTILITY	SYSTEM	MOVIE	MOVIE/CMD	356	200	125
" "	"	"	"	363	204	127
SIDE 2						
MATRIX MANIPULATIONS	LI/4K	-	MATRIX/LV1	18	10	5
" "	"	-	"	88	49	24

SAVE A PACKET ON MICRO-80's DISK DRIVE PACKAGES FOR TRS-80 MODEL 1 AND SYSTEM 80 MICROCOMPUTERS



SINGLE DRIVE PACKAGE *from* ... $499



DUAL DRIVE PACKAGE *from* ... $874

Bigger volume means lower cost price, which we are passing on to you. Avoid the annoying bundle of cables, wires and separate boxes. MICRO-80 is now offering our well-proven MPI disk drives in attractive, self-contained single or dual-drive cabinets complete with internal power supply. Our drive Ø and dual-drive packages also include the appropriate version of DOSPLUS and dual-drive cable.

The best news of all is the specially reduced package prices ...
SAVE $23 — $107 over our already low prices!

Choose the appropriate system from the table below:

DRIVE TYPE	No. of Tracks	No. of Heads	Capacity	Dosplus Version	Price	* Saving
DRIVE Ø						
1 x MPI B51	40	1	100K	3.3	$499	$77.95
1 x MPI B52	40	2	200K	3.4	$639	$97.95
1 x MPI B92	80	2	400K	3.4	$799	$107.95
DRIVE 1						
1 x MPI B51	40	1	100K	—	$415	$23.00
1 x MPI B52	40	2	200K	—	$525	$23.00
1 x MPI B92	80	2	400K	—	$695	$23.00

*Represents the saving compared with buying all the items included in the package separately

•Drive Ø package includes one bare disk drive, self-contained single-drive cabinet/power supply as illustrated, two drive cable and the version of DOSPLUS indicated.

•Drive 1 package includes one bare disk drive and self-contained single-drive cabinet/power supply as illustrated.

If it's a dual-drive system you need, then take advantage of our dual-drive package and
SAVE a further $40 on the price of two single-drive packages ...

DRIVE TYPE	No. of Tracks	No. of Heads	Capacity	Dosplus Version	Price
2 x MPI B51	40 ea	1 ea	2 x 100K	3.3	$874
2 x MPI B52	40 ea	2 ea	2 x 200K	3.4	$1125
2 x MPI B92	80 ea	2 ea	2 x 400K	3.4	$1454

Dual-drive package includes two bare disk drives, self-contained dual-drive cabinet/power supply as illustrated, two drive cables and the version of Dosplus indicated.

NOTE: All 40 track drives are completely compatible with 35 track operating systems such as TRSDOS. DOSPLUS allows you to realise an additional 14% capacity compared with TRSDOS. Under DOSPLUS 3.4, 80 track drives can read 35/40 track diskettes.

All disk drive components are still available separately:

BARE DRIVES — MPI drives offer the fastest track-to-track access time (5 milliseconds) available. All drives are capable of operating in double density for 80% greater storage capacity.

	Price	Freight
MPI B51 40 track, single-head, 100K	$349	$5.00
MPI B52 40 track, dual-head, 200K	$449	$5.00
MPI B92 80 track, dual-head, 400K	$619	$5.00
Separate, dual-drive power supply	$85	$8.00

	Price	Freight
Self-contained, single drive cabinet/power supply	$99	$5.00
Self-contained, dual-drive cabinet/power supply	$135	$5.00
Two drive cable	$39	$2.00
Four drive cable	$49	$2.00
DOSPLUS 3.3	$99.95	$2.00
DOSPLUS 3.4	$149.95	$2.00

Prices are FOB Adelaide. Add $5.00 freight for single drive package, $10.00 for dual-drive package. Prices are in Australian dollars. Freight is road freight anywhere in Australia.

All items carry a 90-day parts and labour warranty. Repairs to be carried out in our Adelaide workshops.

MICRO-80

LEVEL 2 ROM

ASSEMBLY LANGUAGE TOOLKIT

by Edwin Paay

FOR TRS-80 MODEL 1, MODEL 3 AND SYSTEM 80/VIDEO GENIE

This is a new package consisting of two invaluable components:

- **A ROM REFERENCE** Manual which catalogues, describes and cross-references the useful and usable ROM routines which you can incorporate into your own machine language or BASIC programs.
- **DBUG**, a machine language disassembling debugging program to speed up the development of your own machine language programs. DBUG is distributed on a cassette and may used from disk or cassette.

Part 1 of the ROM REFERENCE manual gives detailed explanations of the processes used for arithmetical calculations, logical operations, data movements etc. It also describes the various formats used for BASIC, System and Editor/Assembly tapes. There is a special section devoted to those additional routines in the TRS-80 Model 3 ROM. This is the first time this information has been made available, anywhere. Differences between the System 80/Video Genie are also described. Part 1 is organised into subject specific tables so that you can quickly locate all the routines to carry out a given function and then choose the one which meets your requirements.

Part 2 gives detailed information about each of the routines in the order in which they appear in the ROM. It describes their functions, explains how to use them in your own machine language programs and notes the effect of each on the various Z80 registers.

Part 2 also details the contents of system RAM and shows you how to intercept BASIC routines. With this knowledge, you can add your own commands to BASIC, for instance, or position BASIC programs in high memory — the only restriction is your own imagination!

The Appendices contain sample programmes which show you how you can use the ROM routines to speed up your machine language programs and reduce the amount of code you need to write.

DBUG: Eddy Paay was not satisfied with any of the commercially available debugging programs, so he developed his own. DBUG: allows you to single-step through your program; has a disassembler which disassembles the next instruction before executing it or allows you to bypass execution and pass on through the program, disassembling as you go; displays/edits memory in Hex or ASCII; allows Register editing; has the ability to read and write System tapes and all this on the bottom 3 lines of your screen, thus freeing the rest of the screen for program displays. Four versions of DBUG are included in the package to cope with different memory sizes.

The best news of all is the price. The complete Level 2 ROM ASSEMBLY LANGUAGE TOOLKIT is only:

— Aus. $29.95 + $2.00 p&p
— UK £18.00 + £1.00 p&p

SPECIAL OFFER TO OWNERS OF THE LEVEL II ROM REFERENCE MANUAL ...
UPGRADE TO THIS ASSEMBLY LANGUAGE TOOKIT FOR ONLY $19.95!
Send back your original Level II ROM Reference Manual plus a cheque, money order or Bankcard authorisation for $19.95 plus $2.00 p&p and we will send you the new ASSEMBLY LANGUAGE TOOLKIT

MICRO-80